NPS-CS-09-002

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

Center for Joint Services Electronic Warfare Technical Report

A COMPARISON OF NONLINEAR FILTERS AND MULTI-SENSOR FUSION FOR TRACKING BOOST-PHASE BALLISTIC MISSILES

by

Kyungsu Kim
Phillip E. Pace
Robert G. Hutchins
James B. Michael

January 2009

Approved for public release; distribution unlimited.

Prepared for: U.S. Missile Defense Agency

1

THIS PAGE INTENTIONALLY LEFT BLANK

NAVAL POSTGRADUATE SCHOOL
Monterey, California 93943-5000

Daniel T. Oliver
President

Leonard A. Ferrari
Executive Vice President and
Provost

This report was funded in part by U.S. Missile Defense Agency and the Agency for Defense Development (ADD), South Korea.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE January 2009	3. REPORT TYPE AND DATES COVERED Technical report	
4. TITLE AND SUBTITLE: A COMPARISON OF NONLINEAR FILTERS AND MULTI-SENSOR FUSION FOR TRACKING BOOST-PHASE BALLISTIC MISSILES			5. FUNDING NUMBERS MD7080101P0630
6. AUTHOR(S) Kyungsu Kim, Phillip E. Pace, Robert G. Hutchins, James B. Michael			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Center for Joint Services Electronic Warfare Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER NPS-CS-09-002
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) MISSILE DEFENSE AGENCY Washington, DC			10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES The views expressed in this report are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE A
13. ABSTRACT (maximum 200 words) This report studies two aspects of tracking ballistic missiles during boost phase. The first part compares the performance of several nonlinear filtering algorithms in tracking a single target: the extended Kalman filter (EKF); the unscented Kalman filter (UKF); the particle filter (PF); and, the particle filter with UKF update (UPF). Measurements are range, azimuth and elevation. In the absence of measurement error, all algorithms work well except for the PF, which does not converge. With measurement noise (standard deviations of 10 meters and 1 degree) the EFK also performs poorly, while the UPF is the top performer (although it is also the most computationally intensive). The second part compares the extended information filter (EIF) with earlier work on track scoring to perform sensor/data fusion in a multi-hypothesis framework. Here we find that the EIF handily outperformed other fusion algorithms based on track scoring that we tested.			
14. SUBJECT TERMS Boost-phase Missile Defense, Impulse Modeling, IMPULSE, RF sensors, Multiple Hypotheses Tracking, Extended Kalman Filtering, Unscented Kalman Filtering, Particle Filtering, Unscented Particle Filtering, Multi-sensor fusion, Extended Information Filter			15. NUMBER OF PAGES 93 16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Unlimited

NSN 7540-01-280-5500

Standard Form 298 (Rev 2-89)
Prescribed by ANSI Std 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

A COMPARISON OF NONLINEAR FILTERS AND MULTI-SENSOR FUSION FOR TRACKING BOOST-PHASE BALLISTIC MISSILES

Kyungsu Kim
Phillip E. Pace
Robert G. Hutchins
James B. Michael

Center for Joint Services Electronic Warfare and the Department of Electrical and Computer Engineering

at the

NAVAL POSTGRADUATE SCHOOL
January 2009

ABSTRACT

This report studies two aspects of tracking ballistic missiles during boost phase. The first part compares the performance of several nonlinear filtering algorithms in tracking a single target: the extended Kalman filter (EKF); the unscented Kalman filter (UKF); the particle filter (PF); and, the particle filter with UKF update (UPF). Measurements are range, azimuth and elevation. In the absence of measurement error, all algorithms work well except for the PF, which does not converge. With measurement noise (standard deviations of 10 meters and 1 degree) the EFK also performs poorly, while the UPF is the top performer (although it is also the most computationally intensive).

The second part compares the extended information filter (EIF) with earlier work on track scoring to perform sensor/data fusion in a multi-hypothesis framework. Here we find that the EIF handily outperformed other fusion algorithms based on track scoring that we tested.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

The objective of this study is to investigate various algorithms for tracking multiple ballistic missiles within the boost phase. The ballistic missile behavior and flight profile are simulated using a proprietary add-on MATLAB module, specifically, IMPULSE©. Sensor modeling has been extensively discussed in previous work[16][20] and these same models are used here.

In the first part, the estimation performance of the following nonlinear filters is compared: the extended Kalman filter (EKF), the unscented Kalman filter (UKF), particle filter (PF), and the particle filter with UKF proposal (UPF). Tracking ballistic missiles using infrared (bearing only) and radar (range and bearing) sensors requires a nonlinear filtering approach. The most significant nonlinearity results from the conversion of sensor measurements of range, azimuth and elevation into Cartesian coordinate estimates of the target state. This study compares the relative performances of various non-linear filtering methods applied to tracking multiple ballistic missile targets using radar measurements. The purpose of including the EKF in this study is due to the wide acceptance and frequent application of the EKF. The reason for including the UKF in this study is due to the rapidly growing support for the UKF over the EKF in many applications of nonlinear state estimation. The motivation behind including the PF is its appliance to many nonlinear estimation problems in recent research studies.

Many recent papers have studied the ballistic missile tracking problem. Farian, et al. studied the problem of tracking a ballistic object in the reentry phase using the EKF, UKF and PF in [1] . They found that the results favored the EKF. Bruno and Pavlov also applied a variation of the PF to the case of a reentering ballistic object in [2]. Both of these studies used only a second order coordinate system involving position and velocity, and did not consider acceleration. Saulson and Chang used the UKF, EKF and Central Difference Filter (CDC) to track a ballistic missile in flight from a variety of platforms in [3]. Although they looked at the boost phase, their treatment considered a missile with constant acceleration. Siouris etal. [4] also performed an analysis of the incoming ballistic object using an extended interval Kalman Filter. The American Association of Scien-

tists looked at the Airborne laser problem in the boost phase but only used the EKF for tracking [5]. In addition, they did not consider tracking from the aircraft and the required sensor accuracy. Clemens and Chang examined the use of various nonlinear estimation filters, EKF, UKF, PF, GSPF (Gaussian Sum Particle Filter), for tracking a ballistic missile during boost phase from a moving airborne platform [10].

In the second part, we propose a hierarchical data fusion architecture and method , when a number of different types of sensors are deployed in the vicinity of a ballistic missile launch. Sensor data fusion is the process of combining outputs from several sensors with information from other sources, such as information processing blocks, databases or knowledge bases, into one representational form. It is expected to achieve improved accuracy and more specific inferences than could be achieved by the use of a single sensor alone. In principle, fusion of data from several sensors provide significant advantages over single source data. In addition to the statistical advantage gained by combining same-source data, the use of multiple types of sensors may increase the accuracy with which a quantity can be observed and characterized. In this study, seven active ground-based radar sensors are used to track the ballistic missile in the boost phase. We consider the Information Filter which is an efficient form to fuse the information from multiple sensors without information loss. To evaluate the performance of the Extended Information Filter (EIF) [12], we compare with the track score function proposed in Patsikas's previous work [20] in which the calculation of a track scoring function is used to identify the sensor with the best track file. All simulations have been implemented using MATLAB.

THIS PAGE INTENTIONALLY LEFT BLANK

II. SENSOR MODELS AND MULTIPLE HYPOTHESES TRACKING

In order to detect the threat of a ballistic missile attack, there are a multitude of sensors available to provide early warning information if such an event were to occur. Two particular types of sensors are appropriate for the boost-phase missile tracking problem; passive sensors such as IR, and active sensors such as radar. In this chapter, we investigate surface-based sensors. Furthermore, the mathematical relationships, which determine the performance of each sensor, will be examined. The results will enable us to take the IMPULSE© missile flight information and introduce appropriate error as a means to model the sensed position information as reported by each sensor. In this section, the radar model and multiple hypotheses tracking (MHT) algorithm, originally developed in [16], are reviewed.

A. GENERATING BALLISTIC MISSILE PROFILES

This research makes use of the IMPULSE© simulation tool for generating ballistic missile flight profiles which will later serve as the test data for the tracking algorithm. This software is a product of the National Air & Space Intelligence Center (NASIC)—an organization recognized as the cognizant analysts' representative for threat-missile platforms. IMPULSE© is used by engineers and researchers to conduct missile guidance testing, targeting studies and further enable users to study classified ballistic missile performances. In our study, this software enables us to include flight profiles more sophisticated than simple, constant-rate, parabolic motion-models. The missiles in the simulation display flight performances that one would expect a real-world counterpart to exhibit. IMPULSE© generates missile flight performances to reflect realistic missile physical dynamics, e.g., non-linear velocity and acceleration profiles due to changing mass, staging events, and interaction with the Earth's atmosphere and gravitational field. Figure 2 depicts six missiles, generated with the IMPULSE© software, which have been fired from suspected North Korean missile facilities.

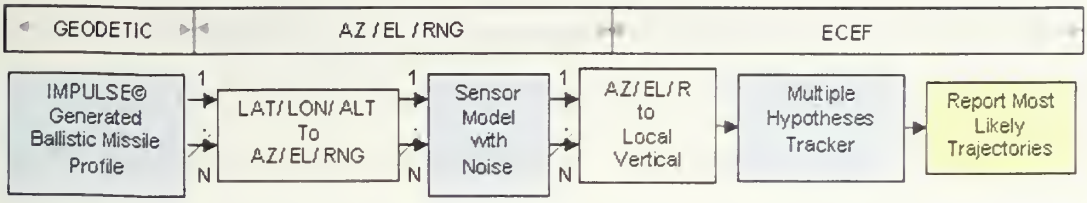

Figure 1. Main functional area implemented in this study to include the IMPULSE@ threat profile, the RF sensor model, and the multiple hypothesis tracking algorithm.

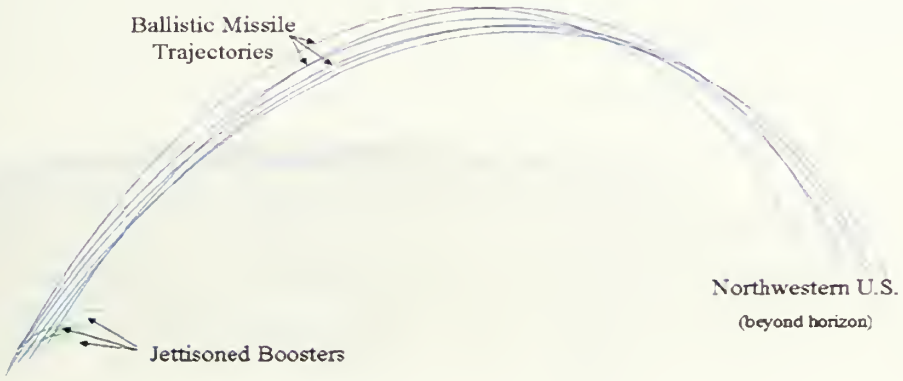

Figure 2. Multiple missile attack directed toward the American continent.

The files listed in Table 1 contain the IMPULSE© output of each missile's flight as exported by the General Write Utility (GWU). Information regarding the use of this tool can be found in Appendix A. These files will serve as the input data to the sensor models and tracking algorithm.

Missile Profile	File
Ballistic Missile 1 Upper Stage data	*Flt1UStage.txt*
Ballistic Missile 1 Lower Stage data	*Flt1LStage.txt*
Ballistic Missile 2 Upper Stage data	*Flt2UStage.txt*
Ballistic Missile 2 Lower Stage data	*Flt2LStage.txt*
Ballistic Missile 3 Upper Stage data	*Flt3UStage.txt*
Ballistic Missile 3 Lower Stage data	*Flt3LStage.txt*
Ballistic Missile 4 Upper Stage data	*Flt4UStage.txt*
Ballistic Missile 4 Lower Stage data	*Flt4LStage.txt*
Ballistic Missile 5 Upper Stage data	*Flt5UStage.txt*
Ballistic Missile 5 Lower Stage data	*Flt5LStage.txt*

Ballistic Missile 6 Upper Stage data	*Flt6UStage.txt*
Ballistic Missile 6 Lower Stage data	*Flt6LStage.txt*

Table 1.

Table 1. Output files from IMPULSE General Write Utility.

Table 2 details the file format for each text file listed in Table 1. The simulation time is given in the first column. The latitude and longitude are represented in the second and third columns. The forth and fifth columns give the altitude in meters and the thrust magnitude in Newtons, respectively.

Time, s	Geodetic Latitude, rad	Geodetic Longitude, rad	Altitude, km	Thurst Magnitude, N
0	0.70969	2.2372	0.02	2.03E+05
1	0.70969	2.2372	0.027182	2.03E+05
2	0.70969	2.2372	0.048894	2.03E+05
3	0.70969	2.2372	0.085382	2.03E+05
4	0.70970	2.2372	0.13561	2.03E+05
5	0.70970	2.2372	0.20043	2.04E+05
6	0.70970	2.2372	0.27988	2.04E+05
7	0.70971	2.2372	0.37395	2.04E+05
8	0.70971	2.2372	0.48265	2.04E+05
9	0.70972	2.2372	0.60599	2.04E+05
10	0.70973	2.2372	0.74394	2.05E+05

Table 2. Column format for each Ballistic Missile file listed in Table 1 as the output from the General Write Utility GUI.

The ballistic missile positions in these files are given in the WGS84 geodetic co-ordinate system. Furthermore, the latitude and longitude values in these files are in units of radians with respect to the center of the Earth. The data can be easily converted to North/South, Degrees/Minutes/Seconds (N/S DD/MM/SS) and East/West, Degrees/Minutes/Seconds (E/W DD/MM/SS) format by using the *rad2deg.m* function in the MATLAB mapping toolbox.

B. SENSOR MODELS

In this study, seven radars, which are placed at different locations relative to the launch position, use the same configuration in order to make the analysis easier to verify. The single pulse, signal to noise ratio (SNR) for each radar system can be expressed as:

$$S/N = \frac{P_t G_t^2 \tau \lambda^2 \sigma}{(4\pi)^3 k T_s B R^4} \tag{0.1}$$

where $T_s = T_a + T_e$, T_s is the system temperature, T_e is the receiver noise temperature and T_a is the antenna noise temperature, which in the work reported here, is assumed to be 290K as was the assumption in [16]. P_t denotes the peak transmitted power, G_t is the antenna gain, σ is the radar cross section of the object , τ is the compressed pulse width, λ is the wavelength, B is the bandwidth of the receiver, k is Boltzmann's constant and R is the range to the target. Generally, the radar losses are also considered but are assumed to be zero for this study. The sensor-to-target slant range, R, is obtained in the simulation by using the MATLAB *elevation* function. This returns the actual target-to-sensor range. Due to the large unambiguous range required, the pulse repetition frequency (PRF) is modeled as PRF = 150Hz. Figure 11 shows the SNR computed at radar sensor 1 (RF1) as it observes the upper and lower stages of the ballistic missile in our simulation. An increasing SNR is observed as the missile passes near and relatively over the sensor during the boost phase of flight. This is observed in the first 100 seconds of the missile's flight. Staging occurs at 65 seconds where a second SNR value—the green line—is initiated.

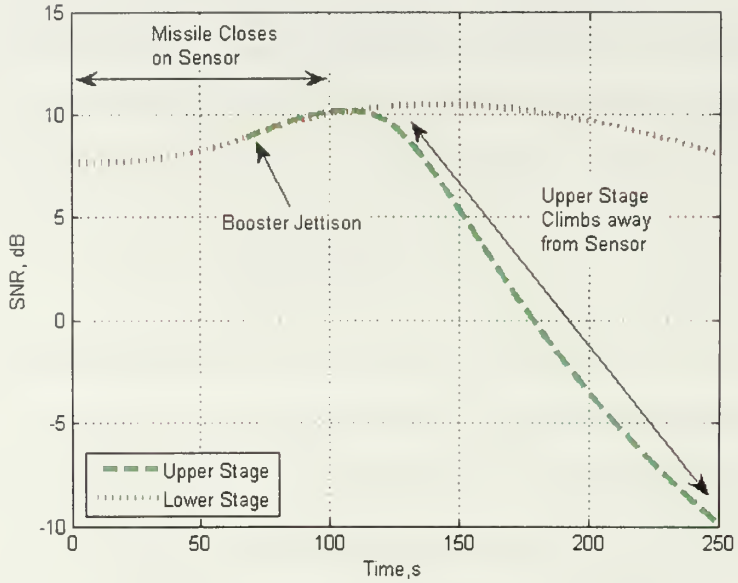

Figure 3. Radar Sensor 1 SNR versus time while observing BM1.

The SNR value observed on the booster—the red plot—remains at a nominal value of 8 dB as it never leaves the sensor's immediate field of view. Conversely, the upper stage of the missile continues its flight away from the sensor. The SNR is inversely proportional to the forth power of the range; as a result, the SNR decays rather rapidly beyond 130 seconds of the simulation. As the SNR changes throughout the course of the missile trajectory, the values will be used in the computation of the error in the reported angular and range measurements.

Table 3 lists the parameters of the radar considered in the simulation.

Parameter	Value
Carrier Frequency, f	10 GHz
Peak Power, P_t	1.0 MW
Antenna Gain, G	42 dB
Radar operating Bandwidth, B	15 MHz
Receiver Noise Temperature, T_e	20

Table 3.
Table 3. RF sensor parameters.

The radar cross section (RCS) of the missile during the flight is calculated by the program POFacets [17]. The missile used in the simulation is a sample unclassified model. To determine RCS of the missile, it is assumed that the missile is a circular cylinder with a radius of 0.60 cm and a length of 20m.

The angle θ between the missile's trajectory vector and the distance vector from the radar to missile, which is required in order to define the RCS function, is determined by the MATLAB function RFobservation by using the geometry of Figure 4. From Figure 4, it is inferred that the angel of incidence is computed as the inner product between the position vector between the sensor and the missile and the tangent vector to the missile trajectory curve. The vector of the curve is defined from the difference of the successive points of the trajectory. The RCS determination is a 3D problem. In this simulation, it is assumed that the change of the angle takes place only on the x-y axes without taking the third dimension into account.

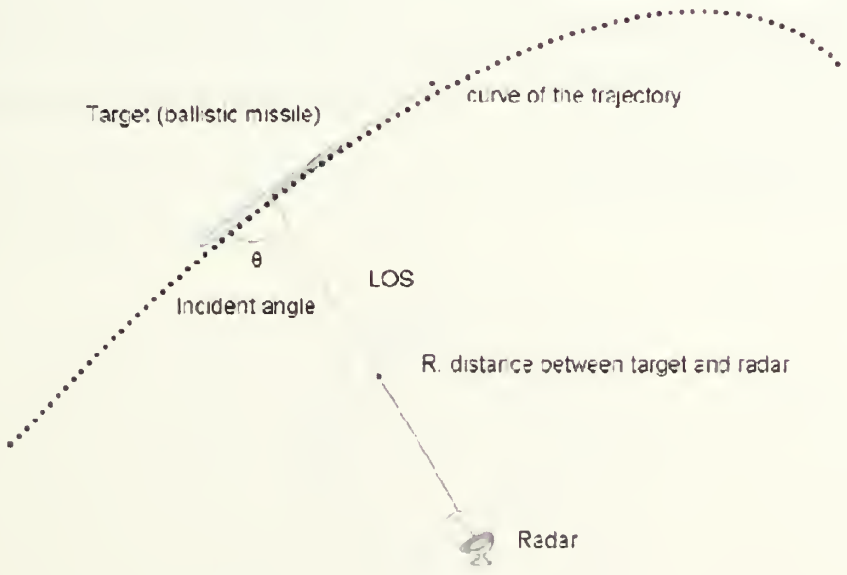

Figure 4. Calculation of the incident angle θ.

Using the above geometry, the MATLAB function RFobservation calculates the incident angle θ for the first radar sensor. A plot of this angle data is shown in Figure 5.

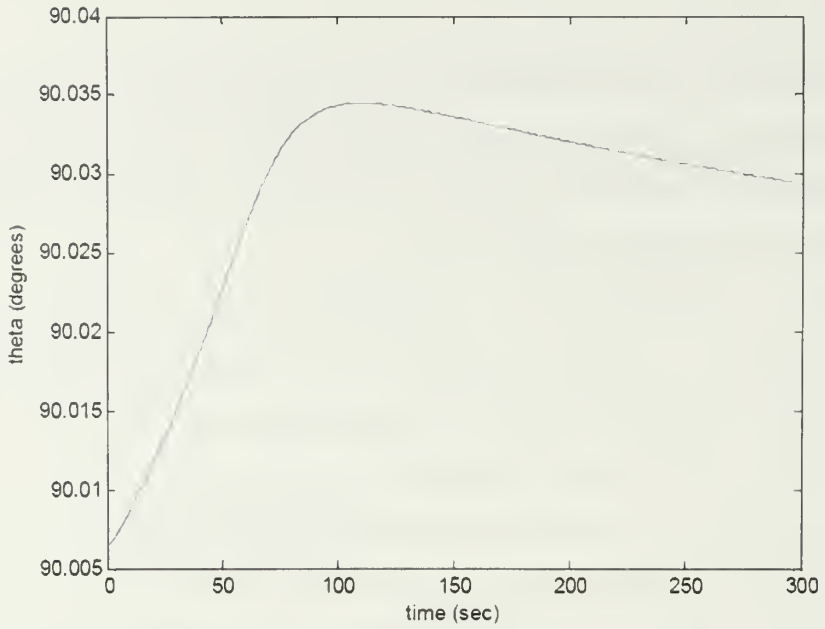

Figure 5. Calculated incident angle θ.

Having determined the incident angle and using the POFacets code, the RCS of the ballistic missile is obtained as shown in Figure 6 and 7.

Figure 6 is a polar plot of the RCS as a function of the angles of incidence. In the vicinity of angle $\theta = 90^0$, which has been calculated from the previous analysis, the values of the RCS are around 36 dBsm.

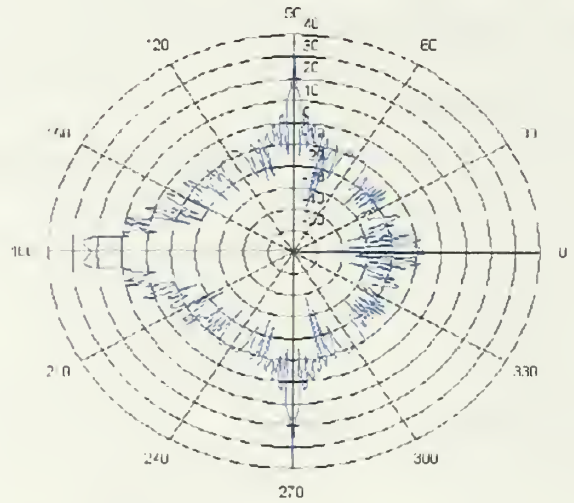

Figure 6. Polar plot of the Ballistic Missile RCS.

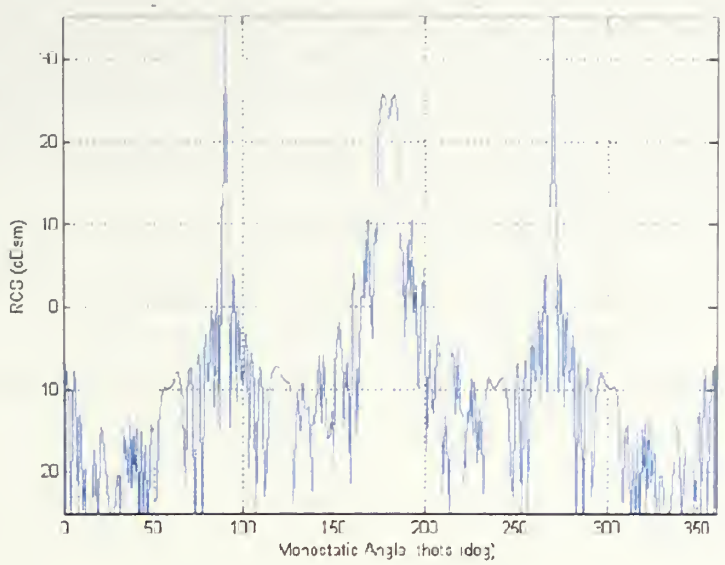

Figure 7. Linear plot of the Ballistic Missile RCS

Figure 7 is a plot of the ballistic missile RCS with respect to the angle of incidence. From the above two plots, it is inferred that the RCS for an angle of incidence in the vicinity of 90^0 is equal to 36 dBsm.

C. MULTIPLE HYPOTHESES TRACKING

In this section, we briefly describe the MHT (Multiple Hypotheses Tracking) algorithm developed in [16].

1. Contacts, Targets, Scans and Associations

To explain multiple target tracking, several terms must be established to provide clarity. First, the definition of a *contact* is introduced. This is an observation that consists of a detection—by a sensor—and its corresponding *measurement*. Commonly, a detection occurs when the signal-to-noise ratio of a sensor meets or exceeds a predefined threshold. The RF signal return from the missile was such that it enabled the sensor to detect the object. Furthermore, the *measurement* corresponding to this detection consists of the position of the object. In limiting the sensor responses to contacts, a restriction is imposed such that a signal return is of interest only when the object meets a predeter-

mined criterion. This allows the contact to be "flagged" for further examination. For the remainder of this study, the term *measurement* will also be used interchangeably to mean contact. These terms, as well as the next several terms given, are illustrated in Figure 8 for a single target tracking problem.

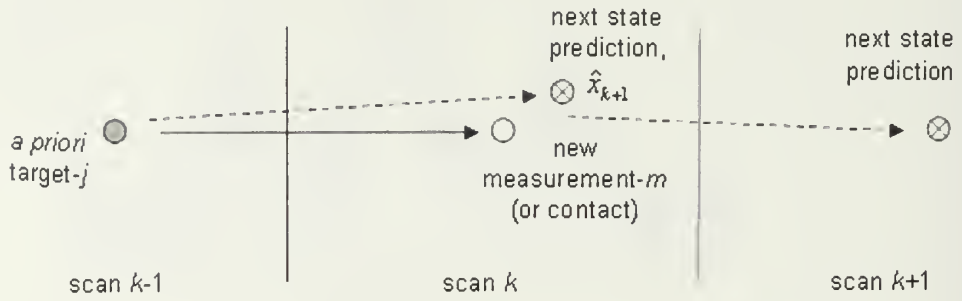

Figure 8. Single target tracking (in 2-D) illustrating measurement-to-target state prediction pairing, and next-state prediction correction.

A *target* is a *measurement* from a previous sample time, which has been flagged and declared an object of interest; hence, the term, *a priori target*, is used. This object's information, or track file, may consist of position, velocity and acceleration parameters. Certain characteristics of a measurement are "screened" and, if they fall within predetermined values, the *measurement* is declared a *target* and stored in a database to await further information update. A contact's velocity information, for instance, may be used to mark the measurement for further observation.

The term *scan* will be used to refer to successive time periods. Allowable measurements and targets will now be further restricted to specific time scans. Finally, the term *association* is used to describe the pairing of a measurement, in a present scan, with that of a target in a previous time scan. It is important to note that there must be no ambiguity in a target-to-measurement pairing from one time period to the next time period.

There are two essential assumptions made in a single target tracking problem. First, there is one and only one target present in the observation region. Secondly, all sensor observations in successive scans are generated by the same target. Given the *a priori* target, j, as in Figure 8, the problem is approached as follows: based on prior information about the target, a *next-state* prediction is generated. This is given by

$$\hat{x}_{k+1} = F_k x_k + \omega_k$$

where x_{k+1} is target next-state vector (prediction), F_k is the known state transition matrix, ω_k is the plant-noise associated with the target and k is the time index. Next, a measurement is taken in the ensuing scan. Since the measurement is likely to originate from the target that caused the contact in the previous scan, a direct comparison may be made between the new observed *contact* and the expected *next-state* prediction. The difference between the observed measurement and the expected prediction is referred to as the *innovation*

The algorithm which computes the next-state prediction is updated with the new observed information. Adjustments are made to improve follow-on estimations. The algorithm eventually stabilizes such that each next-state prediction approaches the actual measurement. Obviously, tracking is straight-forward when there is no uncertainty about the origin of successive measurements and when targets do not accelerate in unpredictable ways.

On the other hand, in a multiple-target environment, for any given scan, the sensor responses—measurements—may be due to any target. Thus, each possible movement of established targets must be hypothesized in the observation space. The primary difficulty in multiple target tracking is correctly assigning successive contacts to a corresponding established target's next state prediction as shown in Figure 9. This process of making possible assignments is referred to as *measurement-to-target* association.

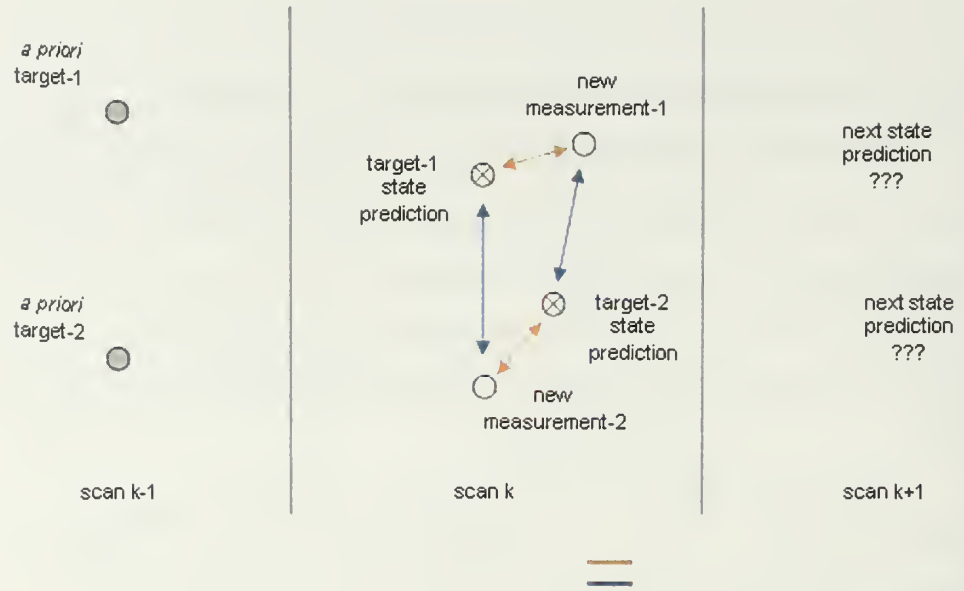

Figure 9. Multiple-target tracking (2-D) scenario. An illustration of the difficulty in correctly pairing new measurements with their state prediction so subsequent predictions converge on the true track.

This is an ideal place to introduce the term association hypothesis in which each hypothesis attempts to provide a likely explanation as to the source of each new measurement. Hence, we have the algorithm's namesake—MHT.

An association hypothesis maps each measurement to a possible target's next-state prediction. Furthermore, if the association of contacts were always obvious, the problem would simplify to independent single target problems—the recursive process of measurement, state prediction, observation, and, finally, update of each target. However, due to the unpredictable nature of each target in a multiple-target space, the associations are ambiguous.

2. MHT Implementation

The following paragraphs cover the implementation of the MHT using the terms and concepts that have been established in Section A. Furthermore, the strategy for solving the association problem is presented in detail.

- Generation of Association Hypotheses

The first step in the measurement-to-target association process is to form multiple feasible hypotheses. The assumptions made in this study are as follows:

- Each target causes, at most, one measurement to be generated by the sensor. Clustering as presented in [21] is an example whereby multiple contacts with common measurements can be represented by a single contact.
- Each measurement corresponds to only one known target. The possibility for sensor false alarms is not addressed in this study.
- Sensor measurements are updated every second.
- Targets, in this case ballistic missiles, do not deploy chaff or use electronic countermeasures.

The following discussion is based upon [21] and provides the framework for the implementation of the MHT. Let $Z^k = \{z_{k,m},\ m = 1,2,...,m_k\}$ denote the set of measurements m in scan time k. In the simulation, a single measurement $z_{k,m}$ can be further decomposed such that $z_{k,m} = \{\xi_{k,m}, \gamma_{k,m}, \zeta_{k,m}\}$ where ξ, γ, and ζ denote the position of the measurement of interest in a predefined coordinate system. In the study of ballistic missiles, a local vertical coordinate system—north, east and up—centered about the sensor's location will be used. Let $Z^k = \{Z^1, Z^2, Z^3, ..., Z^k\}$ represent the cumulative set of measurements up to scan time k. Let $\Omega^k = \{\Omega_i^k,\ i = 1, 2, ..., I\}$ represent the set of all measurement-to-target hypotheses at scan k. The index, i, denotes the hypothesis number. This is the association of measurement in a current scan with *a priori* targets established in preceding scans. Furthermore, the hypothesis Ω_i^k of the kth scan is taken as the joint hypothesis formed from all prior hypotheses Ω_i^{k-1} and the association hypothesis for the current data scan. Using Figure 19 as an illustration, each hypothesis comprises two possible association pairings.

The probability of each feasible hypothesis may now be derived. Let P_i^k be the probability of hypothesis, Ω_i^k, given measurements up through time k. To further clarify, allow Ω_1^k to represent *hypothesis*-1 where *measurement*-1 is associated to *target*-1 (implied by pairing with state *estimate*-1*)* and, within the same hypothesis, *measurement*-

2 is associated to *target*-2 (see Figure 9). The probability of this hypothesis being the correct pairing is represented by P_1^k. Alternatively, *hypothesis*-2, Ω_2^k, with probability P_2^k corresponds to the alternate hypothesis where *measurement*-1 is associated with *target*-2 while *measurement*-2 originates from *target*-1. Also, let P_g^{k-1} represent the probability Ω_g^{k-1}; the past relationships. To successfully perform multiple hypothesis tracking where computational time must remain minimal, a recursive relationship must be established between P_i^k and P_g^{k-1} to avoid reevaluating all past data whenever a present scan set, Z^k, is received.

The association probability, P_i^k, is a conditional probability of Ω_g^k given the set of measurements, Z^k, and the hypothesis, Ω^k. In [22], Nagarajan, Chidambara, and Sharma (NCS) give the relationship

$$P_i^k = P\left(\Omega_g^k \mid Z(k), \Omega^k\right) = P\left(\{\Omega_g^{k-1}, \psi_h\} \mid Z^k, \Omega^k\right)$$

where $\psi_h = \left\{\psi_{mj_h},\ m = 1, 2, 3, ..., m_k\right\}$ and ψ_{mj_h} represents the event that the mth measurement corresponds (originated from) to the jth target as per association hypothesis ψ_h. Furthermore, g denotes the global index while h denotes the hypothesis index. By using Bayes' rule, we can express the above equation as

$$P_i^k = \frac{1}{P\left(\Omega^k\right)} P\left(\Omega_g^{k-1}, \psi_h \mid Z^k\right)$$

By setting $P\left(\Omega^k\right) = C$, a constant, the expression becomes

$$P_i^k = \frac{1}{C} P\left(\Omega_g^{k-1}, \psi_h \mid Z^k\right)$$

which can be further written as

$$P_i^k = \frac{1}{C} P\left(\Omega_g^{k-1}\right) P\left(\psi_h \mid \Omega_g^{k-1}, Z^k\right)$$

where we have again applied the Bayes' rule. Since the events ψ_{mj_h} that comprise ψ_h are independent, we have

$$P_i^k = \frac{1}{C} P\left(\Omega_g^{k-1}\right) \prod_{m=1}^{m_k} P\left(\psi_{mj_h} \mid \Omega_g^{k-1}, Z^k\right)$$

The right-most term is made possible by observing that the current scan association is affected only by Z^k and all past data have been included through the term Ω_g^{k-1}. From [22], by defining $\beta\left(m, j_h, \Omega_g^{k-1}\right) = P\left(\psi_{mj_h} \mid \Omega_g^{k-1}, Z^k\right)$, we have

$$P_i^k = \frac{P\left(\Omega_g^{k-1}\right) \prod_{m=1}^{m_k} \beta\left(m, j_h, \Omega_g^{k-1}\right)}{C} \tag{4.1}$$

The term $\beta\left(m, j_h, \Omega_g^{k-1}\right)$ represents the probability that measurement m corresponds to target j_h as per the present scan hypothesis ψ_h and the past scan hypothesis Ω_g^{k-1}.

To solve for the above probability, let J be the number of known *a prior* targets, and let $j_{k-1,1}, j_{k-1,2}, \ldots, j_{k-1,J}$ have predicted next-state measurement vectors $\hat{x}_{k,j1}, \hat{x}_{k,j2}, \ldots, \hat{x}_{k,jN}$ and corresponding innovation covariance matrices $\Sigma_{k,1}, \Sigma_{k,2}, \ldots, \Sigma_{k,N}$, respectively, as per Ω_g^{k-1} retained from the previous scan k-1. When the new measurement $z_{k,m}$ corresponds to a confirmed target j_N whose existence is implied by the prior hypothesis Ω_g^{k-1}, then $\beta\left(m, j_h, \Omega_g^{k-1}\right)$ is characterized by the probability density function [25]

$$N\left(\tilde{z}_{k,m,j}, \Sigma\right) = \frac{1}{(2\pi)^{n/2} \left|\Sigma_{k,m,j}\right|^{1/2}} \exp\left[-\frac{1}{2} \tilde{z}^T_{k,m,j} \Sigma^{-1}_{k,m,j} \tilde{z}_{k,m,j}\right] \tag{4.2}$$

where $\tilde{z}_{m,j}$ is the measurement-to-target innovation, $\left|\Sigma_{k,m,j}\right|$ is the determinant of the innovation covariance, and n denotes the degrees of freedom of the measurement (in this study, $n = 3$ as the sensor's measurements consist of range, azimuth and elevation).

Furthermore, the innovation is given by [23] [24]

$$\tilde{z}_{k,m,j} \equiv z_{k,m} - H_j \hat{x}_{j,k|k-l} \tag{4.3}$$

and represents the difference between the observed measurement, $z_{k,m}$, and the predicted—or expected—measurement $H_j \hat{x}_{j,k|k-1}$. The observation matrix, H_j, is the used to select the elements of $\hat{x}_{j,k|k-1}$ for comparison. The innovation (or residual) covariance given by

$$\sum_{k,m,j} \equiv H_{\Gamma,k,j} P_{j,k|k-1} H_{\Gamma,k,j}^T + R_k \tag{4.4}$$

The quantities $\tilde{z}_{k,m,j}$ and $\sum_{k,m,j}$ above are obtained from the output of the extended Kalman (EKF) filter as outlined in Appendix B, with the update estimate covariance, $P_{j,k|k-1}$,. The noise covariance, R_k, and the observation matrix $H_{\Gamma,k,j}$ are further explained in Equations B.2 and B.3 of the same appendix. The appendix further provides a comprehensive review of the EKF equations and the motion model used to predict the next state position $\hat{x}_j$ of the target . More details are in [16].

In summary, this chapter described the parameter of the RF sensor at sea level used in the simulation and the multiple hypothesis tracking algorithm based on previous work [16][20].

III. NON-LINEAR FILTERS FOR TRACKING

In this section we describe four recursive estimators (filters) to be used for tracking a ballistic object. The EKF, UKF, PF and PF with UKF proposal are applied in this study. A description of each of the filters used is taken in turn in the following sections.

A. EXTENDED KALMAN FILTER (EKF)

The EKF linearizes nonlinear dynamic models using partial derivatives and the expected state vector in order to perform Kalman Filtering with a Taylor Series approximation of the time and/or measurement updates. As a result of the functional linearization, the EKF provides the optimal linear, or Linear Minimum Mean Square Error (LMMSE), solution. It always approximates the probability density to be Gaussian and is therefore not well suited to handle non-Gaussian distributions such as bi-modal or heavily skewed cases. To do this, the EKF uses partial derivatives of the measurement matrix h with respect to the state x in an attempt to linearize the non-linear dynamic observation equations. The derivative is performed at each time step, k, resulting in the Jacobian matrix, designated H(k). Once the partial derivative is found, standard Kalman Filtering procedures are used to determine the predicted state and it's covariance. More details are in the [16].

B. UNSCENTED KALMAN FILTER (UKF)

The Unscented Kalman Filter is a recursive MMSE estimator that addresses some of the approximation issues of the EKF. Because the EKF only uses the first order terms of the Taylor series expansion of the nonlinear functions, it often introduces large errors in the estimated statistics of the posterior distributions of the states. This is especially evident when the models are highly nonlinear and the Taylor series expansion error becomes significant. Unlike the EKF, the UKF does not approximate the non-linear process and observation models, it uses the true nonlinear models and, instead, approximates the distribution of the state random variable. In the UKF the state distribution is still rep-

resented by a Gaussian random variable (GRV), but it is specified using a minimal set of deterministically chosen sample points. These sample points completely capture the true mean and covariance of the GRV, and when propagated through the true nonlinear system, captures the posterior mean and covariance accurately to the 2^{nd} order for any nonlinearity, with errors only introduced in the 3^{rd} and higher orders [8].

The UKF uses the unscented transformation, or the "scaled unscented transformation" implemented in this study, to generate deterministic sigma points to approximate a random variable's probability distribution given a true (or assumed) nonlinear function and the prior mean and covariance. The sigma points are chosen based on the prior mean and respective columns of the matrix square root of the prior covariance. The Choleski factorization used to solve for the upper triangular matrix square root of the expected state covariance requires that the covariance matrix be positive definite. Square-root implementations of the UKF perform more efficiently and without the positive definite requirement. The UKF is similar to Particle Filters in that it generates points about the mean estimate, but requires less computational cost due to deterministic sampling of the sigma points as apposed to the Particle Filter's random "particles". Unlike Particle Filters, the UKF assumes a Gaussian distribution, but it is able to approximate heavy tailed distributions better than the EKF. For the second-order unscented filter, the estimates of the mean and covariance are accurate to the 2^{nd} order in general and to the 3^{rd} order for Gaussian priors. The second-order unscented filter generally requires at least $2n + 1$ sigma points, while the fourth-order unscented filter requires at least $2n^2 + 1$ sigma points and is able to estimate the mean to the 4^{th} order in general. A reduced sigma point implementation has only the minimum requirement that n+1 sigma points be used [9].

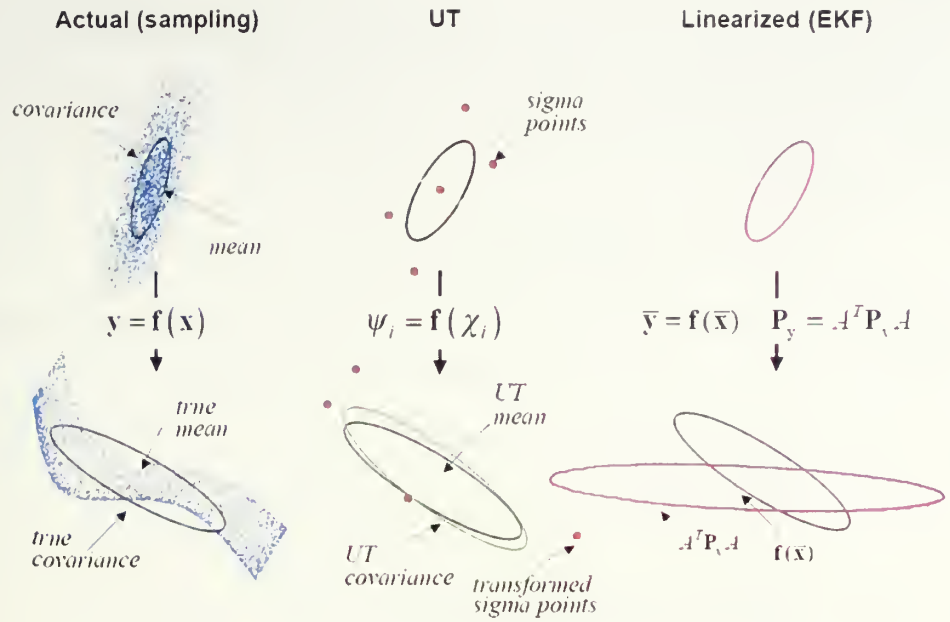

Figure 10. Schematic diagram of the Unscented Transformations.

To calculate the estimation of the state vector x(k+1) using the UKF one begins by developing a matrix χ of 2L+1 sigma vectors χ_i, each with a weight W_i, where L is the dimension of the state vector x (in this case L=9). The sigma vectors and their weights are calculated by

$$\chi_0 = \bar{x}$$

$$\chi_i = \bar{x} - \left(\sqrt{(L+\lambda)P_x}\right)_i \qquad \text{i=1,....L} \qquad (1)$$

$$\chi_i = \bar{x} + \left(\sqrt{(L+\lambda)P_x}\right)_i \qquad \text{i=L+1,...,2L}$$

$$W_0^{(m)} = \lambda/(L+\lambda)$$

$$W_0^{(c)} = \lambda/(L+\lambda) + (1-\alpha^2+\beta)$$

$$W_i^{(m)} = W_i^c = 1/\{2(L+\lambda)\} \qquad \text{i=1,...2L}$$

where $\left(\sqrt{(L+\lambda)P_x}\right)_i$ is the ith row or column of the matrix square root of $(n_x+k)P_x$. The weights are normalized. Each sigma point is propagated through the non-linear function

$$y_i = g(x_i), \quad \text{i=0,...,2L} \tag{2}$$

The estimated mean and covariance of t are computed as the sum of the propagated points

$$\bar{y} = \sum_{i=0}^{2L} W_i y_i \tag{3}$$

$$P_y = \sum_{i=0}^{2L} W_i \left(y_i - \bar{y}\right)\left(y_i - \bar{y}\right)^T$$

Implementation of the above algorithm with the current estimate and its covariance $\hat{x}\langle k \mid k\rangle$ and $P_x\langle k \mid k\rangle$ follows as [1]

1. Compute sigma points ζ_i and their weights W_i, (i=0,...,20) using (7) with α =0.001, β =2, using $\hat{x}\langle k \mid k\rangle$ and. $P_x\langle k \mid k\rangle$. The matrix square root in (1) is found using Cholesky factorization.

2. The sigma points are propagated using the state equation such that

$$\zeta_i\langle k+1 \mid k\rangle = \Theta\left[\zeta_i\langle k \mid k\rangle\right] + Q(k) \tag{4}$$

3. From these new points, compute the predicted state and covariance

$$\hat{x}\langle k+1|k\rangle = \sum_{i=0}^{2L} W_i \zeta_i \langle k+1|k\rangle$$

$$P\langle k+1|k\rangle = \sum_{i=0}^{2L} W_i \left[\zeta_i \langle k+1|k\rangle - \hat{x}\langle k+1|k\rangle\right] \bullet \left[\zeta_i \langle k+1|k\rangle - \hat{x}\langle k+1|k\rangle\right]^T + Q(k) \quad (5)$$

4. Predict the observation sigma points using observation equation

$$\zeta_i \langle k+1|k\rangle = H\left[\zeta_i \langle k|k\rangle\right] \quad (6)$$

5. The observation estimate and covariance follows

$$\hat{y}\langle k+1|k\rangle = \sum_{i=0}^{2L} W_i \zeta_i \langle k+1|k\rangle \quad (7)$$

$$P_{yy}\langle k+1|k\rangle = \sum_{i=0}^{2L} W_i \left[\zeta_i \langle k+1|k\rangle - \hat{y}\langle k+1|k\rangle\right] \bullet \left[\zeta_i \langle k+1|k\rangle - \hat{y}\langle k+1|k\rangle\right]^T + R(k+1) \quad (8)$$

$$P_{xy}\langle k+1|k\rangle = \sum_{i=0}^{2L} W_i \left[\zeta_i \langle k+1|k\rangle - \hat{x}\langle k+1|k\rangle\right] \bullet \left[\zeta_i \langle k+1|k\rangle - \hat{y}\langle k+1|k\rangle\right]^T \quad (9)$$

6. Compute the UKF gain and updated state and state covariance similar to the standard Kalman Filter

$$K(k+1) = P_{xy} P_{yy}^{-1}$$

$$\hat{x}\langle k+1|k+1\rangle = \hat{x}\langle k+1|k\rangle + K(k+1)\left[y(k+1) - \hat{y}\langle k+1|k\rangle\right]$$

$$P\langle k+1|k+1\rangle = P\langle k|k+1\rangle - K(k+1) P_{yy} K^T(k+1) \quad (10)$$

Note that no explicit calculations of Jacobians or Hessians are necessary to implement this algorithm. The UKF requires computation of a matrix square root which can be implemented directly using a Cholesky factorization of order $n_x^3 / 6$. However, the covariance matrices can be expressed recursively, and thus the square-root can be computed in order n_x^2 by performing a recursive update to the Cholesky factorization. So, not only does the UKF outperform the EKF in accuracy and robustness, it does so at no extra computational cost.

C. PARTICLE FILTER (PF)

The previous nonlinear filtering algorithms rely on Gaussian approximations. In this section, we shall present the particle filtering method. It is a set of state estimation methods using a weighted set of samples (particles) in a Monte Carlo methodology to approximate the posterior distribution of the state [10][11]. A large set of particles are generated and weighted to represent the posterior distribution of the state. Each particle is independently propagated through time in accordance with the dynamic state space model and the system process noise. After a new observation y(k+1) arrives, importance evaluation and resampling steps are used to assess the correlation of each individual particle to the observation. New particles are generated to replace the old ones and the particles weights are updated to represent the new posterior. At each step in the process, a sum of the particles is taken to calculate a new state estimate.

This study uses a standard particle filtering algorithm to estimate the state of the missile trajectory. The posterior distribution of the state is assumed to be Gaussian due to the additive Gaussian process noise. The steps to the algorithm are:

1. Generate an initial set of N particles $\left\{x_0^{(i)}; i = 1,...,N\right\}$ from the prior $p(x_0)$ using Gaussian distribution about the estimated location of the missile launch site with zero initial velocity and acceleration n each dimension. Each particle is given an equal normalized weight such that the i^{th} particle weights is $w_0^i = 1/N$, where i=1,…,N. In this study N = 5000, or 10000.

2. For each particle,

(a) Importance sampling step: Perform importance sampling on the particles. A set of likelihood function l_k^i are generated from the difference between the propagated particles and the observation. The corresponding weights are recursively updated based this likelihood function.

$$w_k^{(i)} = \frac{w_{k-1}^{(i)} \square l_k^{(i)}}{\sum_{j=1}^{N} w_k^{(j)}} \tag{11}$$

(b) Selection step (resampling): Resampling of the new weights is accomplished by comparing the particle weights to a threshold. Particles that fall below the threshold are discarded. Those that meet the criteria are resampled to keep the number of particles constant

(c) The new state estimate is then calculated at each time by summing the particle values. Such that

$$\hat{x}_k = \sum_{i=1}^{N} w_k^{(i)} x_k^{(i)} \tag{12}$$

D. UNSCENTED PARTICLE FILTER (UPF)

The unscented Kalman Filter is able to more accurately propagate the mean and covariance of the Gaussian approximation to the state distribution than the EKF. In comparison to the EKF, the UKF tends to generate a more accurate estimate of the true covariance of the state. Distributions generated by the UKF generally have a bigger support overlap with the true posterior distribution than the overlap achieved by the EKF estimates. This is in part related to the fact that the UKF calculates the posterior covariance accurately to the 3rd order, whereas the EKF relies on a first order biased approximation. This makes the UKF a better candidate for more accurate proposal distribution generation within the particle filter framework. The UKF also has the ability to scale the approxima-

tion errors in the higher tailed distributions. Because the sigma point set used in the UKF is deterministically designed to capture certain characteristics of the prior distribution, one can explicitly optimize the algorithm to work with distributions that have heavier tails than Gaussian distributions, i.e. Cauchy or Student-t distributions. This characteristic makes the UKF very attractive for the generation of proposal distributions [7]. The algorithm of the Unscented Particle Filter is followings.

1. Initialization: t = 0

   a. For i = 1, ..., N, draw the states (particle) x_0^i from the prior $p(x_0)$ and set,

$$\bar{x}_0^{(i)} = E\left[x_0^{(i)}\right]$$

$$P_0^{(i)} = E\left[\left(x_0^{(i)} - \bar{x}_0^{(i)}\right)\left(x_0^{(i)} - \bar{x}_0^{(i)}\right)^T\right]$$

$$\bar{x}_0^{(i)a} = E\left[x_0^{(i)a}\right] = \left[\left(\bar{x}_0^{(i)}\right)^T \quad 0 \quad 0\right]^T$$

$$P_0^{(i)a} = E\left[\left(x_0^{(i)a} - \bar{x}_0^{(i)a}\right)\left(x_0^{(i)a} - \bar{x}_0^{(i)a}\right)^T\right] = \begin{bmatrix} P_0^{(i)} & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & R \end{bmatrix}$$

2. For t = 1,2,...

   a. Importance sampling step

      • For i = 1, .., N:

        - Update the particle with the ULF:

          * Calculate sigma points:

$$\chi_{t-1}^{(i)a} = \left[\bar{x}_{t-1}^{(i)a} \quad \bar{x}_{t-1}^{(i)a} \pm \sqrt{\left(n_a + \lambda\right) P_{t-1}^{(i)a}}\right]$$

          * Propagate particle into future (time update):

$$\chi_{t|t-1}^{(i)a} = f\left(\chi_{t-1}^{(i)x}, \chi_{t-1}^{(i)v}\right) \qquad \bar{x}_{t|t-1}^{(i)} = \sum_{j=0}^{2n_a} W_j^{(m)} \chi_{j,t|t-1}^{(i)x}$$

$$P_{t|t-1}^{(i)} = \sum_{j=0}^{2n_a} W_j^{(c)} \left[\chi_{j,t|t-1}^{(i)x} - \bar{x}_{t|t-1}^{(i)}\right]\left[\chi_{j,t|t-1}^{(i)x} - \bar{x}_{t|t-1}^{(i)}\right]^T$$

$$Y_{t|t-1}^{(i)} = h\left(\chi_{t-1}^{(i)x}, \chi_{t-1}^{(i)n}\right) \qquad \bar{y}_{t|t-1}^{(i)} = \sum_{j=0}^{2n_a} W_j^{(m)} Y_{j,t|t-1}^{(i)}$$

* Incorporate new observation (measurement update):

$$P_{\tilde{y}_t\tilde{y}_t} = \sum_{j=0}^{2n_a} W_j^{(c)} \left[Y_{j,t|t-1}^{(i)} - \bar{y}_{t|t-1}^{(i)}\right]\left[Y_{j,t|t-1}^{(i)} - \bar{y}_{t|t-1}^{(i)}\right]^T$$

$$P_{x_t x_t} = \sum_{j=0}^{2n_a} W_j^{(c)} \left[\chi_{j,t|t-1}^{(i)} - \bar{x}_{t|t-1}^{(i)}\right]\left[Y_{j,t|t-1}^{(i)} - \bar{y}_{t|t-1}^{(i)}\right]^T$$

$$K_t = P_{x_t y_t} P_{\tilde{y}_t\tilde{y}_t}^{-1} \qquad \bar{x}_t^{(i)} = \bar{x}_{t|t-1}^{(i)} + K_t\left(y_t - \bar{y}_{t|t-1}^{(i)}\right)$$

$$\hat{P}_t^{(i)} = P_{t|t-1}^{(i)} - K_t P_{\tilde{y}_t\tilde{y}_t} K_t^T$$

- Sample $\hat{x}_t^{(i)} \square q\left(x_t^{(i)} \mid x_{0:t-1}^{(i)}, , y_{1:t}\right) = N\left(\bar{x}_t^{(i)}, \hat{P}_t^{(i)}\right)$

- Set $\hat{x}_{0:t}^{(i)} \square \left(x_{0:t-1}^{(i)}, \hat{x}_t^{(i)}\right)$ *and* $\hat{P}_{0:t}^{(i)}\left(P_{0:t-1}^{(i)}, \hat{P}_t^{(i)}\right)$

• For i =1, … , N, evaluate the importance weights up to a normalizing constant.

b. Selection step

• Multiply/Suppress particles $\left(\hat{x}_{0:t}^{(i)}, \hat{P}_{0:t}^{(i)}\right)$ with high/low importance weights $\tilde{w}_t^{(i)}$, respectively, to obtain N random particles $\left(\tilde{x}_{0:t}^{(i)}, \tilde{P}_{0:t}^{(i)}\right)$

c. MCMC step (optional)

- Apply a Markov transitional kernel with invariant distribution $p(x_{0:t}^{(i)} \mid y_{1:t})$ to obtain $\left(x_{0:t}^{(i)}, P_{0:t}^{(i)}\right)$.

d. Output: The output is generated in the same manner as for the generic particle filter.

IV. SENSOR DATA FUSION

Sensor data fusion is the process of combining outputs from sensors with information from other sensors, information processing blocks, database or knowledge bases, into one representational form. This technique is expected to achieve improved accuracy and more specific inferences than could be achieved by the use of a single sensor alone [14]. The applications of sensor data fusion include automated target recognition, guidance of autonomous vehicles, remote sensing, battlefield surveillance, automatic threat recognition systems, monitoring of manufacturing processes, robotics and medical. The techniques employed in data fusion are drawn from diverse disciplines like digital signal processing, statistical estimation and control theory and classical numerical methods [14]. In principle, fusion of data from several sensors provides significant advantages over single source data. In addition to the statistical advantage gained by combining same-source data, the use of multiple types of sensors may increase the accuracy with which a quantity can be observed and characterized. Raw data from similar sensors may be directly combined if the sensors are measuring the same physical phenomenon. However feature/state vector or decision level fusion may be employed to combine data that come from non-commensurate sensors.

While attempting to build a sensor data fusion system, the architecture that is chosen for fusion plays a significant role in deciding the algorithm for data fusion. The choice of architecture is made with a view to balancing computing resources, available communication bandwidth, desired accuracy and the capabilities of the sensors [14].

The three commonly used architecture are (i) centralized (ii) distributed/decentralized and (iii) hybrid. The distributed/decentralized architectures have several inherent operational advantages that are dictated by the current trends towards modular and autonomous systems [15]. For such architecture, information filtering which essentially tracks information about states is found to be most suitable. The information filter is found to provide direct interpretation of node observations and contributions in terms of information. Prediction and estimation in terms of information reduce the computational load particularly for nonlinear systems in multi sensor systems.

The linear information filter (LIF) and extended information filter (EKF) are descried here. The information filter is a Kalman filter recast in terms of the information state vector and information matrix. The advantages of this formulation is that the estimation equations are computationally simpler than the measurement update equations of the Kalman filter, although prediction is more complex. Information filter is simpler to decouple and decentralize amongst a network of sensor nodes. A further advantage of the information filter is that it may be partitioned to provide a simple hierarchical estimation architecture based on the communication of the information terms from sensor nodes to a common fusion center and information state estimates from nodes to a common assimilation point. The proposed fusion architecture is also described in the end of this section.

A. INTRODUCTION TO THE INFORMATION FILTER

The diverse and sometimes conflicting information obtained from multiple sensors gives rise to the problem of how the information may be combined in a consistent and coherent manner. This is the data fusion problem. Multisensor fusion is the process by which information from multiple sensors is combined to yield a coherent description of the system under observation. A variety of information based data fusion algorithms have been employed in the recent work of Mutambara [12] and that of Mayika et al [13].

Most current sensor fusion algorithms consider systems described by linear dynamics and observation models, whereas, most practical problems have nonlinear dynamics and sensor information nonlinearly dependent on the states that describe the environment. Information-based estimation makes fully decentralized data fusion for nonlinear systems attainable. Consider a linear system with state-space representation,

$$x(k) = F(k)x(k-1) + B(k)u(k-1) = w(k-1) \tag{1}$$

Where $x(k)$ is the state of interest at time k, $F(k)$ is the state transition matrix form time (k-1) to k while, u(k) and B(k) are the input control vector and matrix, respectively. $w(k) \square N(0, Q(k))$ is the associated process noise modeled as an uncorrelated, zero

mean and white sequence. The system is observed according to the linear discrete equation

$$z(k) = H(k)x(k) + v(k) \tag{2}$$

Where z(k) is the vector of observations made at time k. H(k) is the observation matrix or model and $v(k) \sim N(0, R(k))$ is the associated observation noise modeled as an uncorrelated white sequence. The estimate of the state x(j) at time i given information up to and including time j is given by

$$\hat{x}(i \mid j) = E\left[x(i) \mid z(1), \ldots z(j)\right]$$

$$P(i \mid j) = E\left[\left(x(i) - \hat{x}(i \mid j)\right)\left(x(i) - \hat{x}(i \mid j)\right)^T \mid z(1), \ldots z(j)\right]$$

B. LINEAR INFORMATION FILTER

For a system described by equation (1) and being observed according to Equation (2), the Kalman filter provides a recursive estimate $\hat{x}(k \mid k)$ for the state x(k) at time k given all information up to time k in terms of the predicted state $\hat{x}(k \mid k-1)$ and the new observation z(k). The one-step-ahead prediction, $\hat{x}(k \mid k-1)$, is the estimate of the state at a time k given only information up to time (k-1). The information filter is essentially a Kalman filter expressed in terms of measure of information about the parameters (state) of interest rather than direct state estimate and their associated covariances. This filter has also been called the inverse covariance form of the Kalman Filter. Assuming Gaussian noise and minimum mean squared error estimation gives the Fisher information matrix as,

$$F(k) = p^{-1}(k \mid k)$$

This information matrix or the inverse of the covariance matrix, is central to the filtering techniques. The general description of a linear system is given by:

$$X_{k+1} = \phi X_k + w_k \tag{3}$$

$$Z_k = HX_k + v_k \tag{4}$$

Where X_k is the state vector at time k and ϕ is the transition matrix. The input noise w_k is assumed to be zero mean white Gaussian process with variance Q. The measurement noise v_k is assumed to be zero mean, white Gaussian with variance R. The information state vector y(i|j) and the information matrix Y(i|j) are related to the state vector and the covariance matrix P(i|j) by the equations:

$$\hat{y}(i \mid j) = P^{-1}(i \mid j)\hat{x}(i \mid j) \tag{5}$$

$$\hat{Y}(i \mid j) = P^{-1}(i \mid j) \tag{6}$$

Defining

$$i(k) = H^T(k)R^{-1}z(k) \tag{7}$$

As the information state contribution from an observation z(k), and

$$I(k) = H^T(k)R^{-1}H(k) \tag{8}$$

As its associated information matrix and

$$L(k \mid k-1) = P^{-1}(k \mid k-1)F(k)P(k-1 \mid k-1) \tag{9}$$

As a propagation matrix (independent of the observation made), linear filter is written in terms of the information state/matrix:

Prediction:

$$\hat{y}(k \mid k-1) = L(k \mid k-1)\hat{y}(k-1 \mid k-1) \tag{10}$$

$$\hat{Y}(k \mid k-1) = \left[F(k)\hat{Y}^{-1}(k-1 \mid k-1)F^{T}(k) + Q(k) \right]^{-1} \tag{11}$$

Estimation:

$$\hat{y}(k \mid k) = \hat{y}(k-1 \mid k-1) + i(k) \tag{12}$$

$$\hat{Y}(k \mid k) = \hat{Y}(k \mid k-1) + I(k) \tag{13}$$

The advantage if this formulation is that the estimation equation (12) and (13) are computationally simpler than the measurement update equations of the Kalman filter.

C. EXTENDED INFORMATION FILTER

The linear information filter can now be extended to a linearized estimation algorithm for nonlinear systems by using principles form both the derivations of the Information filter and the EKF. This generates a filter that predicts and estimates information about nonlinear state parameters given nonlinear observations and nonlinear system dynamics. It is the Extended Information filter (EIF) and provides a solution to the nonlinear estimation problem. The EIF also has all the advantages of the Information filter and resolves some of the problems associated with the EKF. The estimation for nonlinear systems, in particular multisensory systems, is best carried out using information variables rather than state variables.

It is important to note that the EIF can't be extrapolated from the Information filter and EKF in an obvious manner. This is because in the nonlinear case, the function of the Information filter depends on this separation, which is possible in the linear observation equation. In a real data fusion problem, the state of interest could evolve in a nonlinear manner, in which case simple linear models cannot adequately describe such a nonlinear system. Also, sensor observations may not be linear functions of the states. The general nonlinear system could be described by the following models.

$$X(k) = f(X(k-1), k) + w(k) \tag{14}$$

$$Z(k) = h(X(k), k) + v(k) \tag{15}$$

Where X(k) is the state vector and w(k) is additive process noise vector at time step k. The nonlinear function f(.,k) is the nonlinear state transition function mapping the previous state to the current state. Z is the observation vector at time k and h is the nonlinear observation model mapping the current state to the observations. The EIF algorithm is the followings:

Prediction:

$$\hat{y}(k \mid k-1) = \hat{Y}(k \mid k-1) f(k, \hat{x}(k-1 \mid k-1)) \tag{16}$$

$$\hat{Y}(k \mid k-1) = \left[\nabla f_x(k) \hat{Y}^{-1}(k-1 \mid k-1) \nabla f_x{}^T(k) + Q(k) \right]^{-1} \tag{17}$$

Estimation:

$$\hat{y}(k \mid k) = \hat{y}(k \mid k-1) + i(k) \tag{18}$$

$$\hat{Y}(k \mid k) = \hat{Y}(k \mid k-1) + I(k) \tag{19}$$

The information state/matrix are given by

$$I(k) = \nabla h_x^T(k) R^{-1}(k) \nabla h_x(k) \tag{20}$$

$$i(k) = \nabla h_x^T(k) R^{-1}(k) [v(k) + \nabla h_x(k) \hat{x}(k \mid k-1)] \tag{21}$$

where v(k) is the innovation sequence given by

$$v(k) = z(k) - h(\hat{x}(k \mid k-1)) \tag{22}$$

The EIF described above has several attractive practical features:

- The filter solves, in information space, the linear estimation problem for systems with both nonlinear dynamics and observations. In addition to having

all the attributes of the Information filter, it is a more practical and general filter.

- The information estimation Equation (18) and (19) are computationally simpler Than the EKF estimation equations. This makes the partitioning of these equations for decentralized systems easy.
- Although the EIF prediction Equation (16) and (17) are of the same apparent complexity as the EKF ones, they will be easier to distribute and fuse because of the orthonormality properties of information space parameters.
- Since the EIF is expressed in terms of information matrices and vectors, it is easily initialized compared to the EKF. Accurate initialization is important where linearized models are employed.

Some of the drawbacks inherent in the EKF still affect the EIF. These include the nontrivial nature of Jacobian matrix derivation (and computation) and linearization instability.

D. DISTRIBUTED AND DECENTRALIZED DATA FUSIOIN SYSTEMS

The nature of data fusion is that there are number of sensors physically distributed around an environment. In a centralized data fusion system, raw sensor information is communicated back to a central processor, where the information is combined to produce a single fused picture of the environment. In a distributed data fusion system, each sensor has it's own local processor which can generally extract useful information from the raw sensor data prior to communication. This has the advantage that less information is normally communicated, the computational load on the central processor is reduced and the sensors themselves can be constructed in a reasonably modular manner. The degree to which local processing occurs at a sensor site varies substantially from simply validation and data compression up to the full construction of tracks or interpretation of information locally.

While for many systems a centralized approach to data fusion is adequate, the increasing sophistication, functional requirements, complexity and size of data fusion systems, coupled with the ever reducing cost of computing power argues more and more toward some form of distributed processing. The central issue in designing distributed data fusion systems is the development of appropriate algorithms which can operate at a number of distributed sites in a consistent manner.

This section begins with a general discussion of data fusion architectures and introduces the data fusion architecture used in this research.

1. Data Fusion Architecture

Distributed data fusion systems may take many forms. At the simplest level, sensors could communicate information directly to a central processor where it is combined. Little or no local processing of information need take place and the relative advantage of having many sources of information is sacrificed to having complete centralized control over the processing and interpretation of this information. As more processing occurs locally, so computational and communication burden can be removed from the fusion center, but at the cost of reduced direct control of low-level sensor information. Increasing intelligence of local sensor nodes naturally results in a hierarchical structure for the fusion architecture. This has the advantage of imposing some order on the fusion process, but the disadvantage of placing a specific and often rigid structure on the fusion system. Other distributed architectures consider sensor nodes with significant local ability to generate tracks and engage in fusion tasks. Fully decentralized architectures have no central processor and no common communication system. In such systems, nodes can operate in a fully autonomous manner, only coordinating through the autonomous communication information.

In a hierarchical structure, the lowest level processing elements transmit information upwards, through successive levels, where the information is combined and refined, until at the top level some global view of the state of the system is made available. Such hierarchical structures are common in many organizations and have many well-known

advantages over fully centralized systems; particularly in reducing the load on a centralized processor while maintaining strict control over sub-processor operations.

The hierarchical approach to systems design has been employed in a number of data fusion systems and has resulted in a variety of useful algorithms for combining information at different levels of a hierarchical structure. The Figures 11 and 12 show hierarchical fusion systems. A hierarchical approach to the design of data fusion systems also comes with a number of inherent disadvantages. The ultimate reliance on some central processor or controlling level within the hierarchical means that reliability and flexibility are often compromised. Failure of this central unit leads to failure of the whole system, changes in the system often mean changes in both the central unit and in all related sub-units. Further, the burden placed on the central unit in terms of combining information can often still be prohibitive and lead to an inability of the design methodology to be extended to incorporate an increasing number of sources of information.

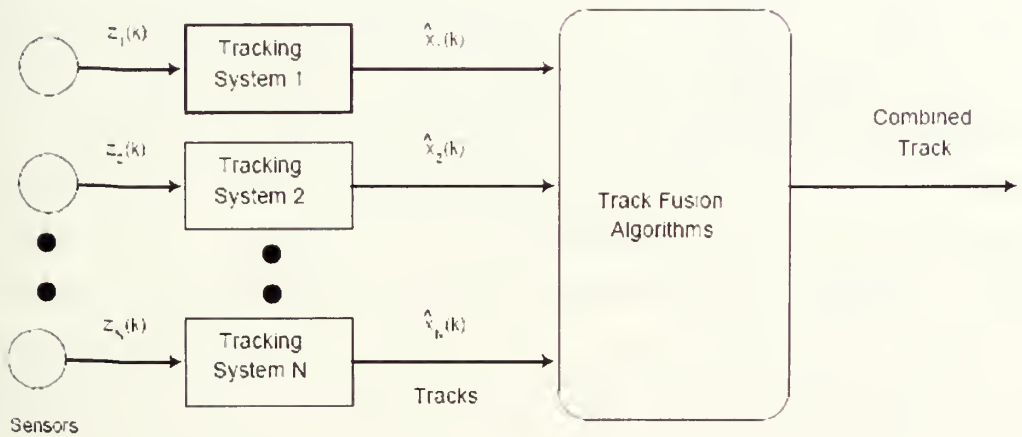

Figure 11. Single Level Hierarchical Multiple Sensor Tracking System

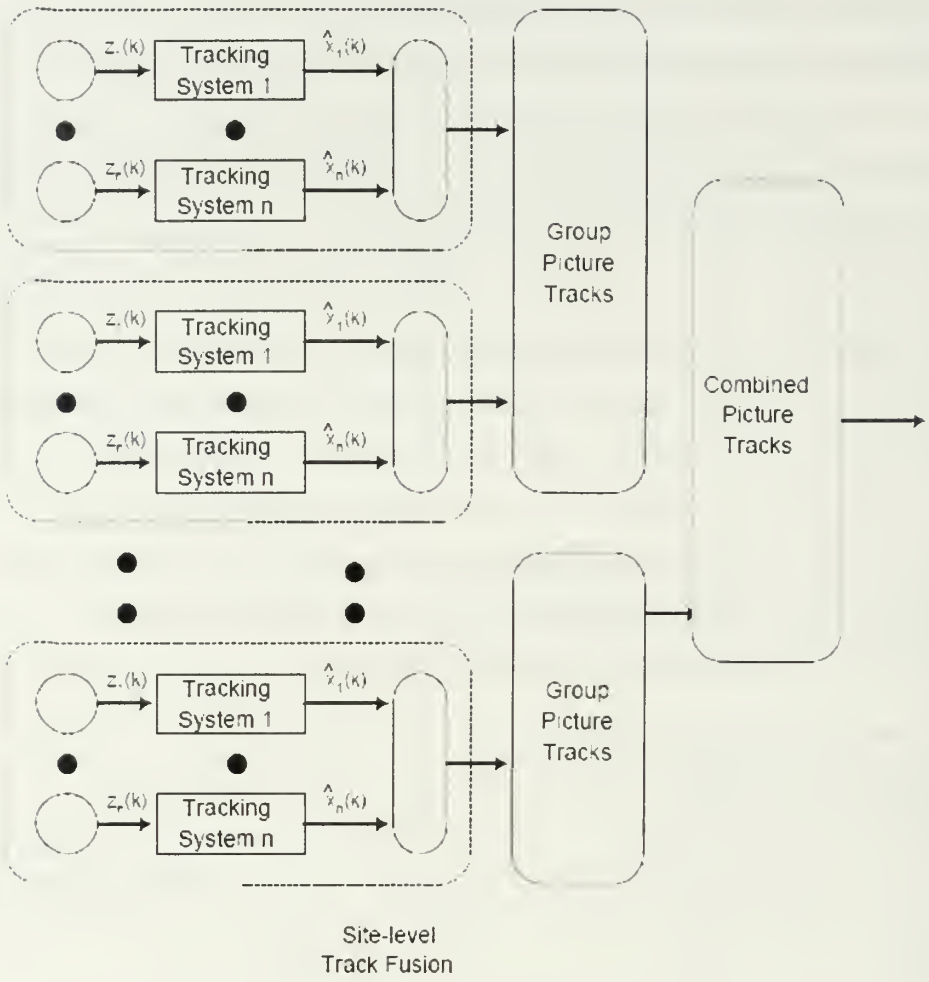

Figure 12. Multiple Level Hierarchical Multiple Sensor Tracking System

The move to more distributed, autonomous, organizations is clear in many information processing systems. This is most often motivated by two main considerations; the desire to make the system more modular and flexible, and a recognition that a centralized or hierarchical structure imposes unacceptable overheads on communication and central computation. The migration to distributed system organizations is most apparent in Artificial Intelligence (AI) application areas, where distributed AI has become a research area in its own right. Many of the most interesting distributed processing organizations have originated in this area.

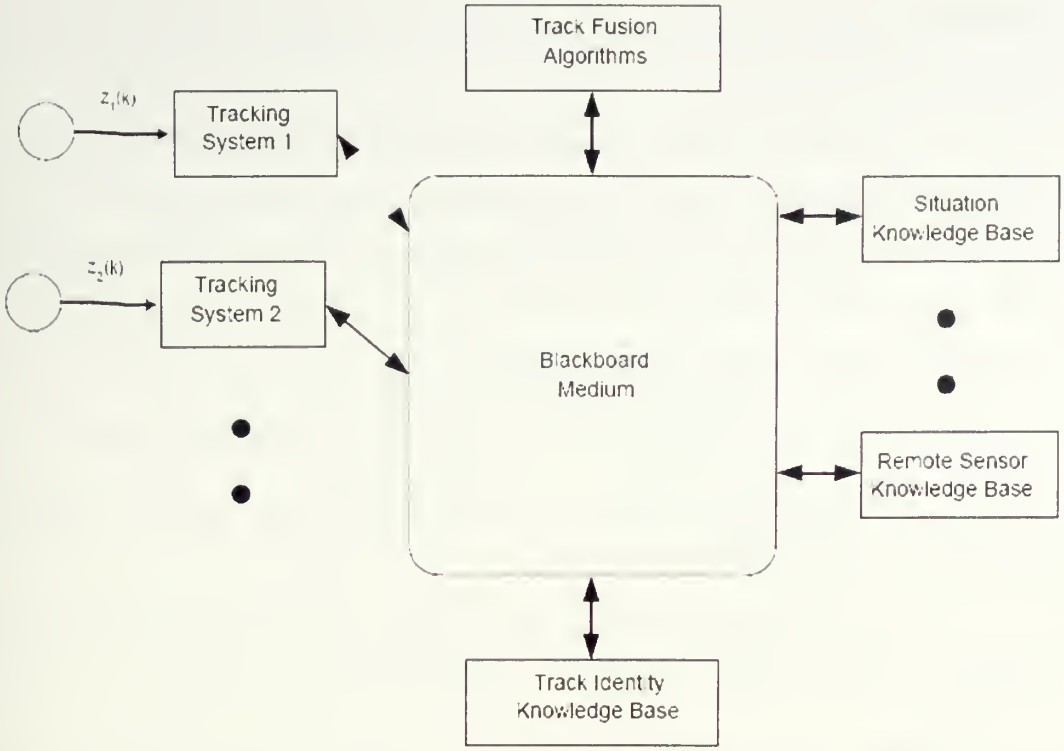

Figure 13. Blackboard Architecture in Data Fusion

The above notable figure is the "Blackboard" architecture, originally developed in the Hearsay speech understanding program, but now widely employed in many areas of AI and data fusion research. A Blackboard architecture consists of a number of independent autonomous "agents". Each agent represents a source of expert knowledge or specified information facility or shared memory resource. This resource is called blackboard. The blackboard is designed to closely replicate its physical analogue. Each agent is able to write information or local knowledge to this resource. Every agent in the system is able to read from this resource, in an unrestricted manner, any information which it considers useful in its current task. In principle, every agent can be made modular and new agents may be added to the system when needed without changing the underlying architecture or operation of the system as a whole. The flexibility of this approach to system organization has made the Blackboard architecture popular in a range of application domains. In data fusion, the Blackboard approach has been most widely used for knowledge-based data fusion systems in data interpretation and situation assessment. However,

the structured nature of tracking and identification problems does not lent itself to this anarchic organizational form.

A decentralized data fusion system consists of a network of sensor nodes, each with its own processing facility, which together do not require any central fusion or central communication facility. In such a system, fusion occurs locally at each node on the basis of local observations and the information communicated from neighboring nodes. At no point is there a common place where fusion of global decisions are made.

A decentralized data fusion system is characterized by three constraints:

- There is no simple single central fusion center; no one should be central to the successful operation of the network.

- There is no common communication facility; nodes cannot broadcast results and communication must be kept on a strictly node-to-node basis.

- Sensor nodes do not have any global knowledge of sensor network topology; nodes should only know about connections in their own neighborhood.

Figure 14 show three possible realizations of a decentralized data fusion system. The key point is that all these systems have no central fusion center.

The constraints imposed provide a number of important characteristics for decentralized data fusion systems:

- Eliminating the central fusion center and any common communication facility ensures that the system is **scalable** as there are no limits imposed by centralized computational bottlenecks or lack of communication bandwidth.

- Ensuring that no node is central and that no global knowledge of the network topology is required for fusion means that the system can be made survivable to the on-line loss (or addition) of sensing nodes and to dynamic changes in the network structure.

- As all fusion processes must take place locally at each sensor site and no global knowledge of the network is required a priori, nodes can be constructed and programmed in a modular fashion.

These characteristics give decentralized systems a major advantage over more traditional sensing architectures, particularly in defense applications.

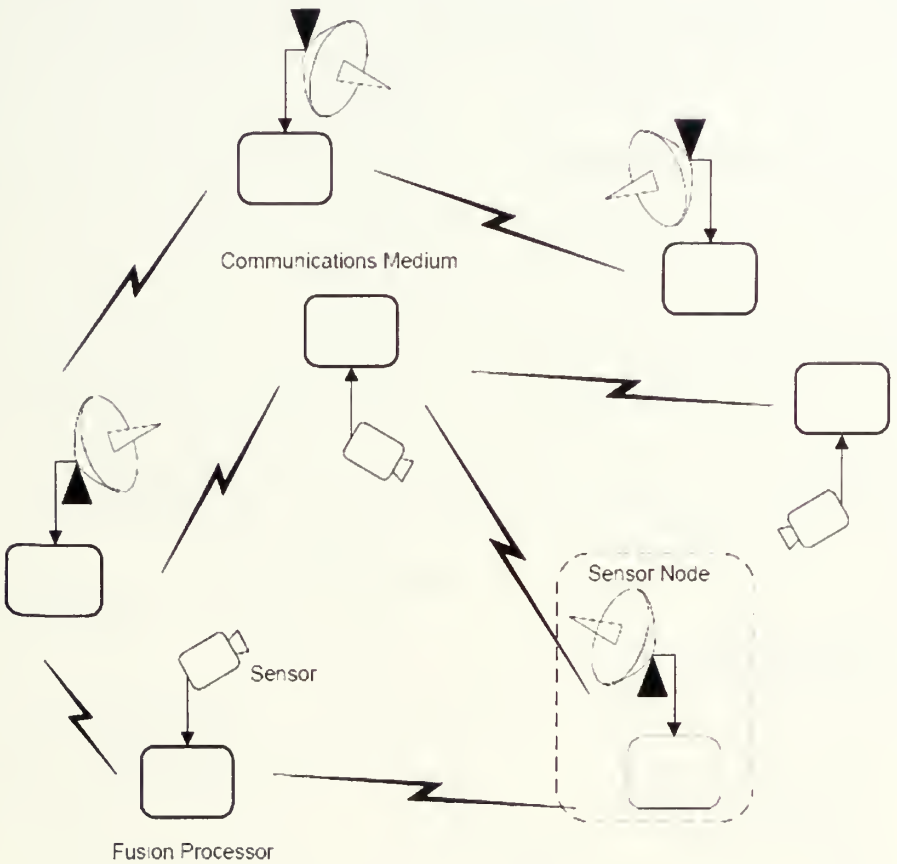

Figure 14. Blackboard Architecture in Data Fusion

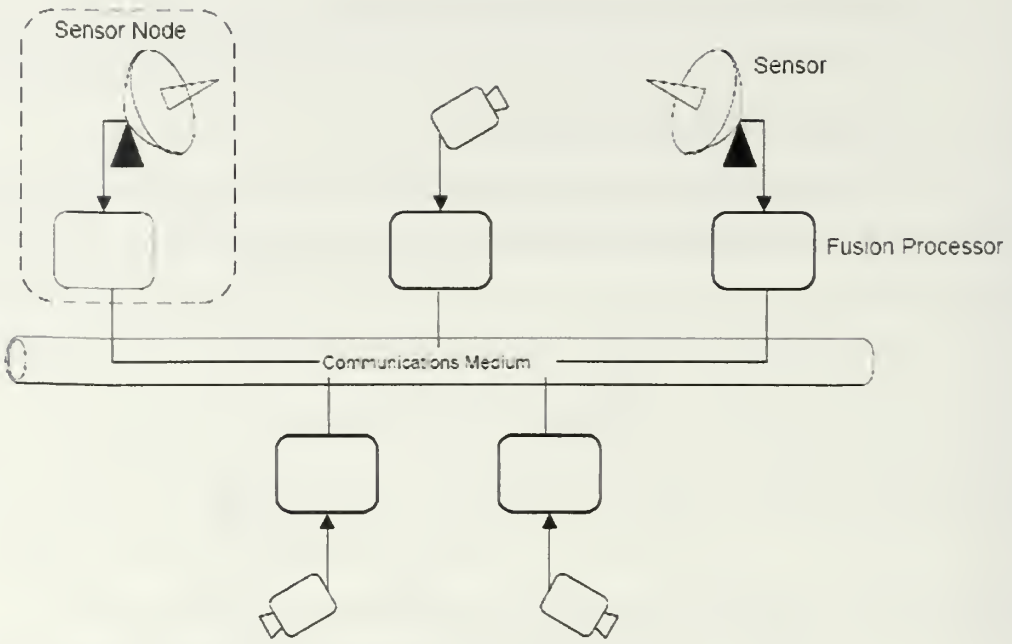

Figure 15. Blackboard Architecture in Data Fusion

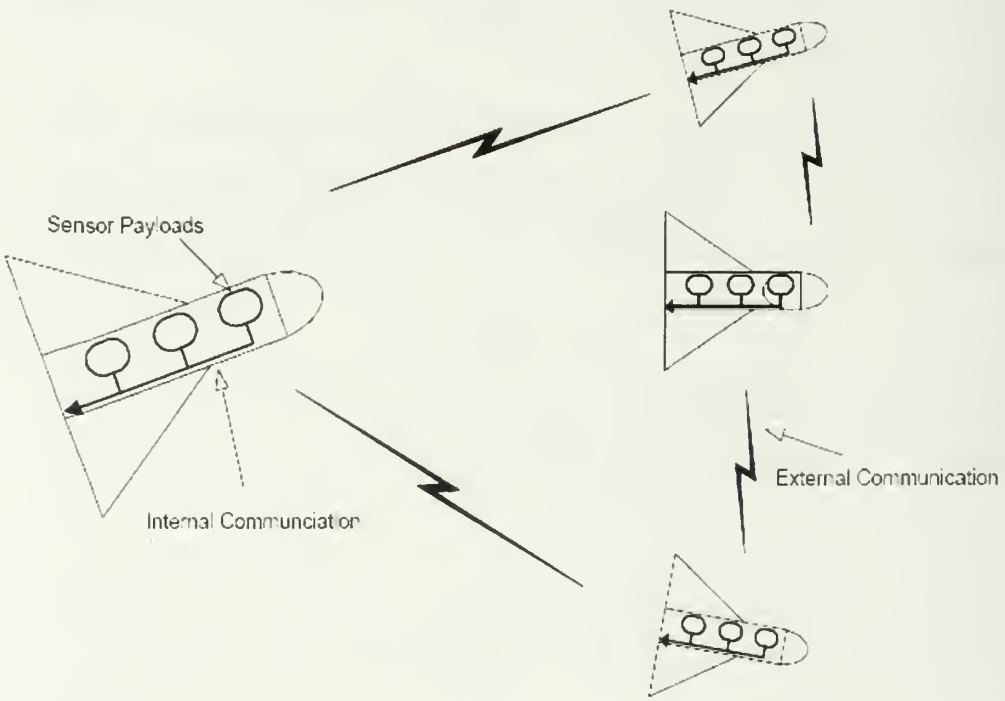

Figure 16. Blackboard Architecture in Data Fusion

2. Proposed Fusion Architecture

The proposed hierarchical fusion architecture is a simple hierarchical estimation architecture based first on the communication of the information terms $i(\square)$ and $I(\square)$ from sensor nodes to a common fusion center. In this system, information-state contributions are calculated at each sensor node and transmitted to a central fusion center where a common estimate is obtained by simple summation. All state predictions are undertaken at the central processor. Each sensor incorporates a full state model and takes observations according to the Equation (2). They all calculate an information state contribution from their observations in terms of $i_i(k)$ and $I_i(k)$. These are then communicated to the fusion center and are incorporated into global estimate through Equations (18) and (19). The information-state prediction is generated centrally using Equations (16) and (17) and the state estimate itself may be found at any stage from $\hat{x}(i \mid j) = Y^{-1}(i \mid j)\hat{y}(i \mid j)$. To avoid communicating predictions to nodes, any validation or data association should take place at the fusion center. Figure 17 shows the hierarchical architecture.

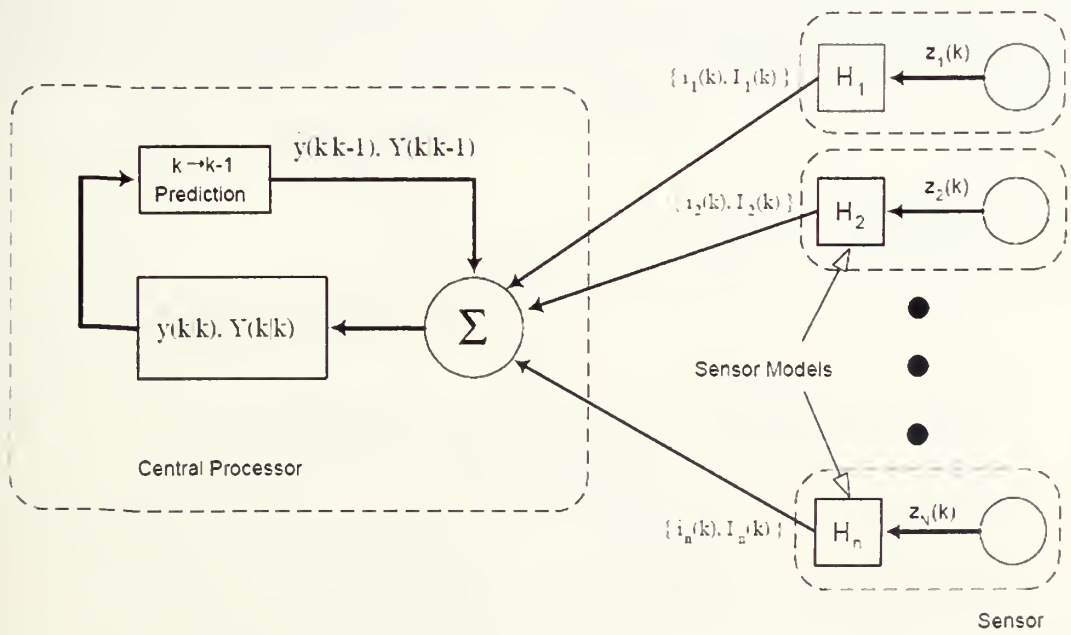

Figure 17. A Proposed simple hierarchical fusion architecture

V. SIMULATION RESULTS

In this section, we will describe the results of simulation studies on the performance of nonlinear filter and sensor fusion using the proposed fusion architecture. The IMPULSE model and MATLAB are the tools used to implement the simulation. Specifically, the results from the IMPULSE program are used by the MATLAB code from [16] in order to simulate the tracking problem. The number of trajectories used in the simulation is totally 12 including upper and lower stage for each of 6 ballistic missile trajectories. A detailed description of the requirements (inputs) and capabilities (outputs) of the IMPULSE program are outlined in [16]. An unclassified model of the ballistic missile is used. Figure 23 shows the trajectories.

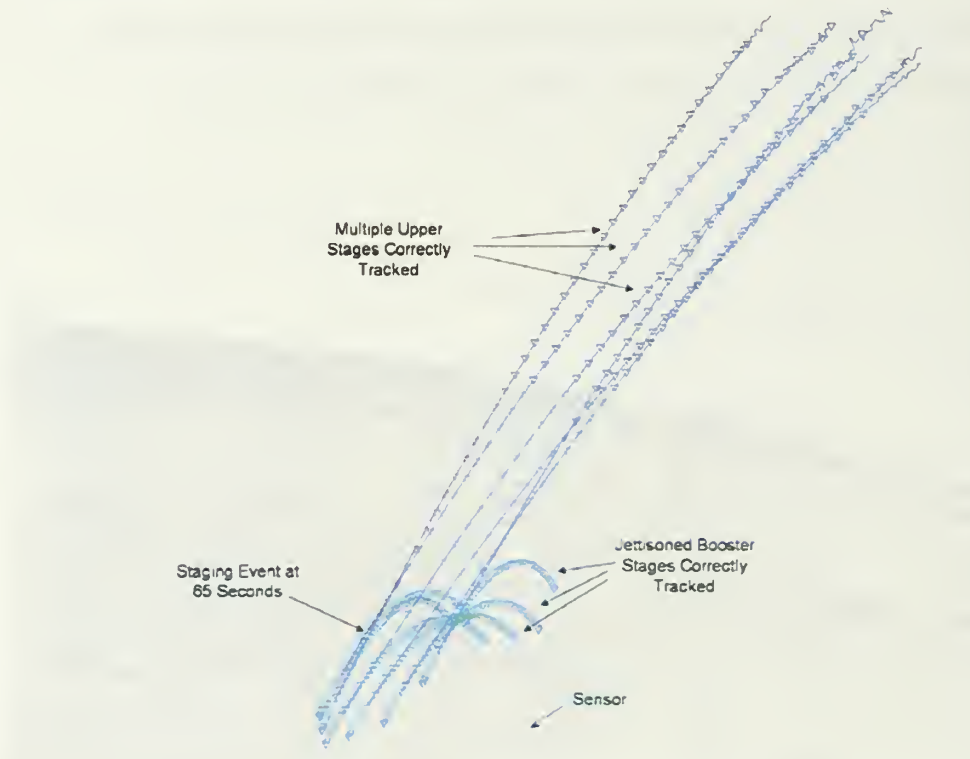

Figure 18. Six near-simultaneous launches of ballistic missiles (viewed looking north). The delay ranges from four to ten second intervals. Missile staging occurs at 65 to 85 seconds, simulation time. The coordinate system here is local vertical (north-east-up) with respect to the sensor.

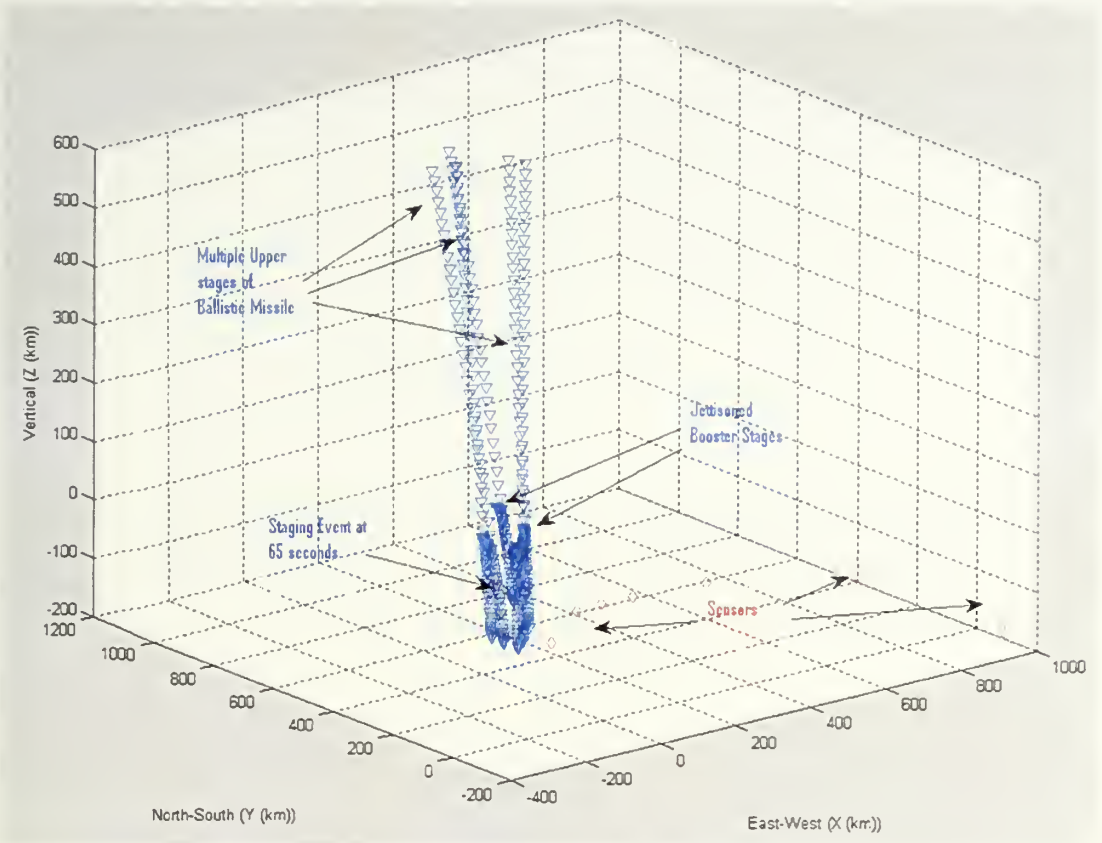

Figure 19. Same profile as in Figure 23: Six, near simultaneous, ballistic missile launches (looking East-to-West) and 7 RF sensors

Table 4 lists the positions of the sensors in the vicinity of the launched ballistic missile. The radars are placed at sea level.

Sensors	Position		
	Latitude	Longitude	Altitude
RF #1	131°46'44"E	40°50'45"N	Sea level
RF #2	132°46'44"E	41°40'45"N	Sea level
RF #3	131°00'00"E	40°00'00"N	Sea level

RF #4	130°00'00"E	35°00'00"N	Sea level
RF #5	135°00'00"E	40°00'00"N	Sea level
RF #6	140°00'00"E	45°00'00"N	Sea level
RF #7	145°00'00"E	50°00'00"N	Sea level

Table 4. Sensor positions

A.MODELS OF TARGET MOTION AND RADAR MEASUREMENTS

A ballistic missile tracking algorithm is EIF based on the extended Kalman form. The system equations are the standard tracking equations

$$\hat{x}_{k+1} = F_k \hat{x}_k + G_k + \omega_k$$

$$z_k = h_k x_k + \nu_k$$

where the missile state vector is given as

$$x_k = \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \\ y \\ \dot{y} \\ \ddot{y} \\ z \\ \dot{z} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix}$$

The term F_k is the linear state transition matrix:

$$F_k = \begin{bmatrix} 1 & \Delta & \frac{\Delta^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta & \frac{\Delta^2}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta & \frac{\Delta^2}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The gravity matrix, G_k, which accounts for the acceleration in $\ddot{z}$ due to gravity with $g = 9.8,\ m \cdot s^{-2}$, is

$$G_k = -g \times \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{\Delta^2}{2} \\ \Delta \\ 0 \end{bmatrix}$$

and ω_k is the plant noise with covariance Q_k

$$Q_k = q^2 \times \begin{bmatrix} \frac{\Delta^5}{20} & \frac{\Delta^4}{8} & \frac{\Delta^3}{6} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\Delta^4}{8} & \frac{\Delta^3}{3} & \frac{\Delta^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\Delta^3}{6} & \frac{\Delta^2}{2} & \Delta & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\Delta^5}{20} & \frac{\Delta^4}{8} & \frac{\Delta^3}{6} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\Delta^4}{8} & \frac{\Delta^3}{3} & \frac{\Delta^2}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\Delta^3}{6} & \frac{\Delta^2}{2} & \Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta^5}{20} & \frac{\Delta^4}{8} & \frac{\Delta^3}{6} \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta^4}{8} & \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta^3}{6} & \frac{\Delta^2}{2} & \Delta \end{bmatrix}$$

where Δ is the sampling interval and q^2 is a scaling factor used to account for unmodeled target maneuver accelerations, and ν_k is the measurement noise with covariance

$$R_k = \begin{bmatrix} \sigma_R^2 & 0 & 0 \\ 0 & \sigma_A^2 & 0 \\ 0 & 0 & \sigma_E^2 \end{bmatrix}$$

Although the missile dynamics in this system are linear, the measurement process is nonlinear. The sensor platform observes the missile positions through measurements in range, azimuth and elevation relative to the sensor.

Let

$$h_k = \begin{bmatrix} \text{Range} \\ \text{Azimuth} \\ \text{Elevation} \end{bmatrix}$$

where

$$\text{range} = \sqrt{x^2 + y^2 + z^2} = \sqrt{x_1^2 + y_4^2 + z_7^2}$$

$$\text{azimuth} = \tan^{-1}\left[\frac{y}{x}\right] = \tan^{-1}\left[\frac{x_4}{x_1}\right]$$

$$\text{elevation} = \tan^{-1}\left[\frac{z}{r'}\right]$$

with

$$r' = \sqrt{x^2 + y^2} = \sqrt{x_1^2 + y_4^2}$$

These measurement equations are nonlinear and therefore must be converted using a series expansion of the measurement equation h_k. Applying the definition of the $H_{\Gamma,k}$ matrix, the gradient of h_k is determined as follows:

$$H_{\Gamma,k} = \begin{bmatrix} \frac{\partial r(x)}{\partial x_1} & \frac{\partial r(x)}{\partial x_2} & \frac{\partial r(x)}{\partial x_3} & \frac{\partial r(x)}{\partial x_4} & \frac{\partial r(x)}{\partial x_5} & \frac{\partial r(x)}{\partial x_6} & \frac{\partial r(x)}{\partial x_7} & \frac{\partial r(x)}{\partial x_8} & \frac{\partial r(x)}{\partial x_9} \\ \frac{\partial a(x)}{\partial x_1} & \frac{\partial a(x)}{\partial x_2} & \frac{\partial a(x)}{\partial x_3} & \frac{\partial a(x)}{\partial x_4} & \frac{\partial a(x)}{\partial x_5} & \frac{\partial a(x)}{\partial x_6} & \frac{\partial a(x)}{\partial x_7} & \frac{\partial a(x)}{\partial x_8} & \frac{\partial a(x)}{\partial x_9} \\ \frac{\partial e(x)}{\partial x_1} & \frac{\partial e(x)}{\partial x_2} & \frac{\partial e(x)}{\partial x_3} & \frac{\partial e(x)}{\partial x_4} & \frac{\partial e(x)}{\partial x_5} & \frac{\partial e(x)}{\partial x_6} & \frac{\partial e(x)}{\partial x_7} & \frac{\partial e(x)}{\partial x_8} & \frac{\partial e(x)}{\partial x_9} \end{bmatrix}$$

which simplifies to

$$H_{\Gamma,k}=\begin{bmatrix} \frac{x_1}{\sqrt{x_1^2+x_4^2+x_7^2}} & 0\ 0 & \frac{x_4}{\sqrt{x_1^2+x_4^2+x_7^2}} & 0\ 0 & \frac{x_7}{\sqrt{x_1^2+x_4^2+x_7^2}} & 0\ 0 \\ \frac{-x_4}{x_1^2+x_4^2} & 0\ 0 & \frac{-x_4}{x_1^2+x_4^2} & 0\ 0 & 0 & 0\ 0 \\ \frac{-x_1\cdot x_7}{\sqrt{x_1^2+x_4^2}\left(x_1^2+x_4^2+x_7^2\right)} & 0\ 0 & \frac{-x_4\cdot x_7}{\sqrt{x_1^2+x_4^2}\left(x_1^2+x_4^2+x_7^2\right)} & 0\ 0 & \frac{\sqrt{x_1^2+x_4^2}}{x_1^2+x_4^2+x_7^2} & 0\ 0 \end{bmatrix}$$

Finally, the approximate—linearized—system of equations for the state predict and measurement are the followings.

$$\hat{x}_{k+1}=F_k\hat{x}_k+G_k+\omega_k$$

$$z_k=H_k x_k+\nu_k$$

B.COMPARISON OF PERFORMANCE OF NONLINEAR FILTERS

In this section, we describe the experimental results of comparing the performance of three nonlinear filters, EKF, UKS, and UPF , since the PF did not converge. We used the distance error, which is the square root of sum for each coordinate distance error, as a measure to evaluate the performance. In this study, two values for the measurement noise are chosen to have the following standard deviations:

- $\sigma_{range}=\sigma_{azimuth}=\sigma_{elevation}=0$
- $\sigma_{range}=10$ meters, $\sigma_{azimuth}=1^{\circ}$, $\sigma_{elevation}=1^{\circ}$

Figures 21-24 show the average distance errors for two of the trajectories in the experiments over 30 Monte Carlo runs. Each sensor performs the estimation for each trajectory and then the results are averaged for the time period. The red line shows the distance error calculated from each coordinates after EKF estimation is done. The green line shows the results for the UKF and the blue line is for the UPF. The diamond marker

is designating the distance error at the last time. As the graphs show, when standard deviations for measurement noise are all set to 0, the EKF is the best.

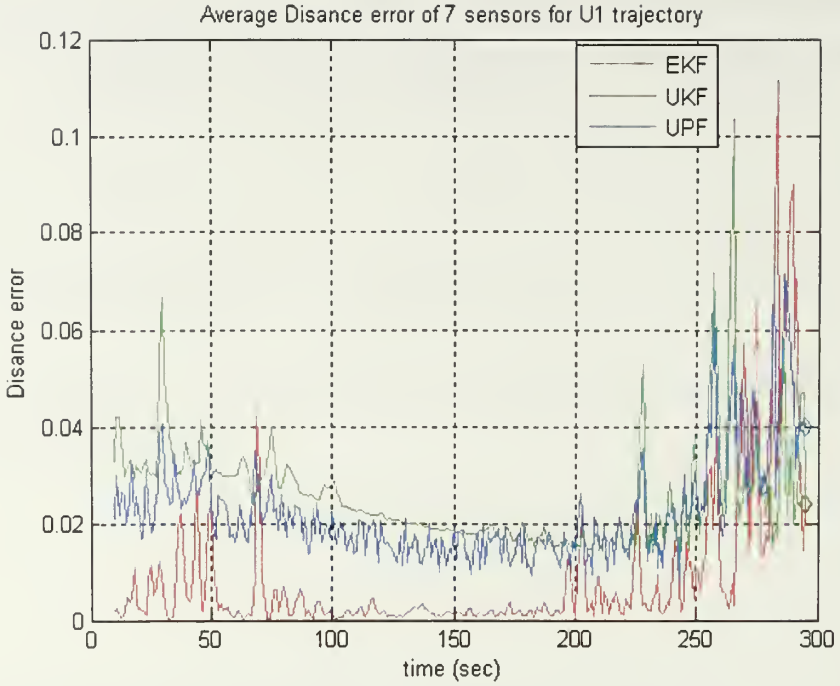

Figure 20. Average distance error of seven sensors for 1st upper stage trajectory ($\sigma_{range} = \sigma_{azimuth} = \sigma_{elevation} = 0$)

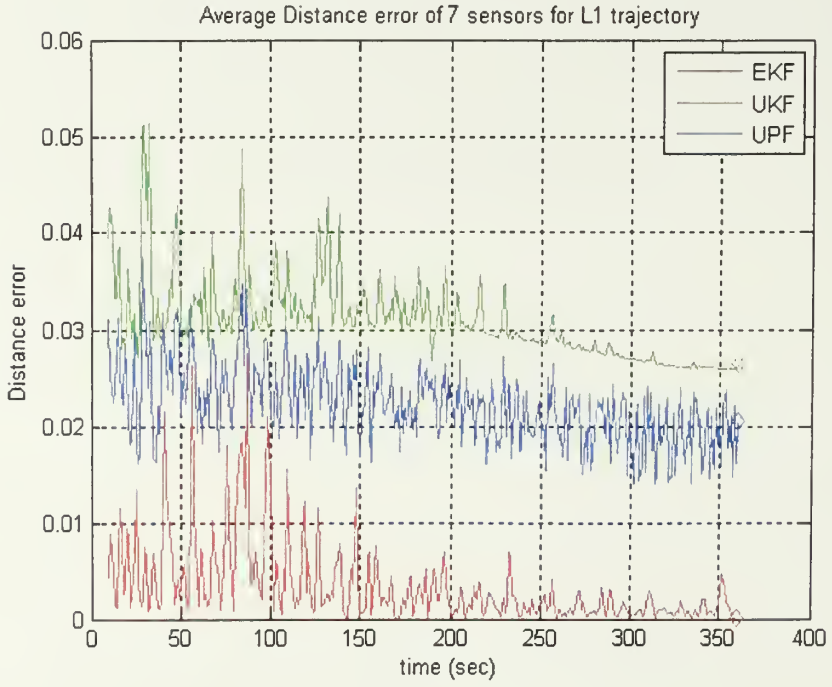

Figure 21. Average distance error of seven sensors for 1st lower stage trajectory ($\sigma_{range} = \sigma_{azimuth} = \sigma_{elevation} = 0$)

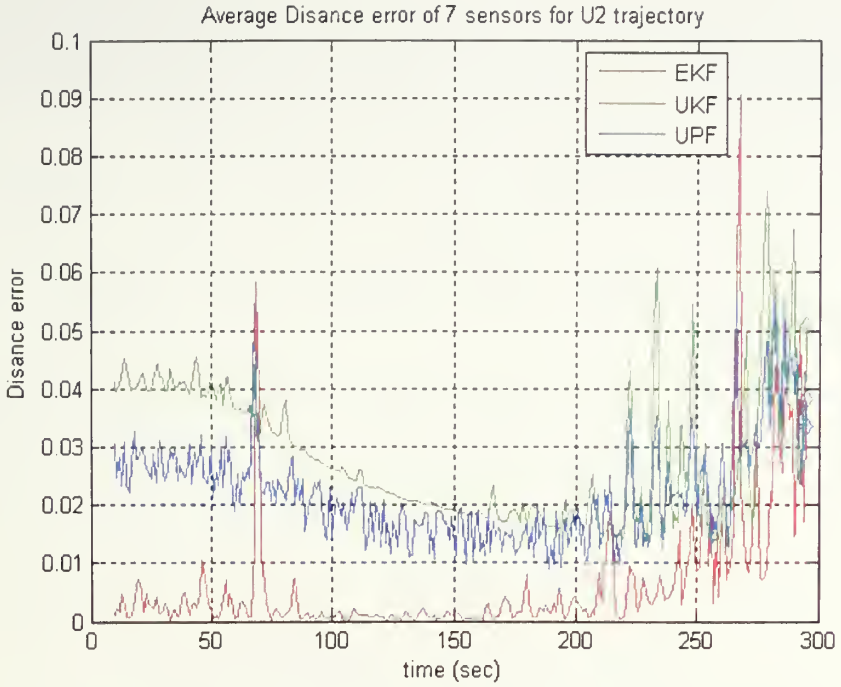

Figure 22. Average distance error of seven sensors for 2nd upper stage trajectory ($\sigma_{range} = \sigma_{azimuth} = \sigma_{elevation} = 0$)

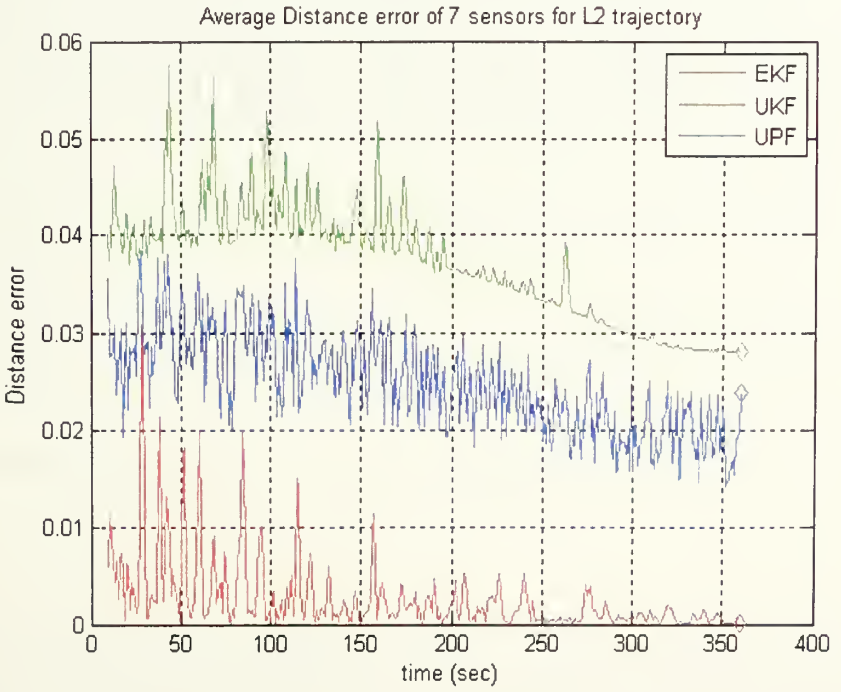

Figure 23. Average distance error of seven sensors for 2nd lower stage trajectory ($\sigma_{range} = \sigma_{azimuth} = \sigma_{elevation} = 0$)

Table 5 shows average distance error and final distance error at last time for all trajectories in number

Trajectory	EKF		UKF		UPF	
	Average	Last	Average	Last	Average	Last
U1	0.0089	0.0238	0.0258	0.0237	0.0223	0.0399
L1	0.0034	0.0001	0.0308	0.0264	0.0222	0.0203
U2	0.063	0.0381	0.0281	0.0506	0.0222	0.0334
L2	0.0026	0.0007	0.037	0.028	0.0247	0.0226
U3	0.0076	0.0231	0.0259	0.0226	0.0214	0.035
L3	0.0036	0.0008	0.0323	0.0283	0.0229	0.0152
U4	0.0166	0.2043	0.0315	0.0988	0.0292	0.0938
L4	0.0013	0.0007	0.029	0.0212	0.0201	0.0142
U5	0.0079	0.0559	0.037	0.0316	0.0273	0.0259
L5	0.0047	0.0006	0.0374	0.033	0.026	0.0175
U6	0.0067	0.0078	0.0282	0.0595	0.0225	0.041
L6	0.0037	0.0006	0.0385	0.0325	0.0261	0.0253

Table 5. Average distance error and distance error at last time of three nonlinear filters for 12 trajectories ($\sigma_{range} = \sigma_{azimuth} = \sigma_{elevation} = 0$)

However, the performance of the EKF is degraded very badly for the second case. Figures 24-27 shows the average distance error for four of the trajectories. The second figure compares the distance error of only UPF and UPF. They work well without the performance degradation for the second case, as the results show. Even though the UPF

has the best performance of three filters, because of its time complexity, we can know that the UKF is the best estimation algorithm for ballistic missile tracking through .

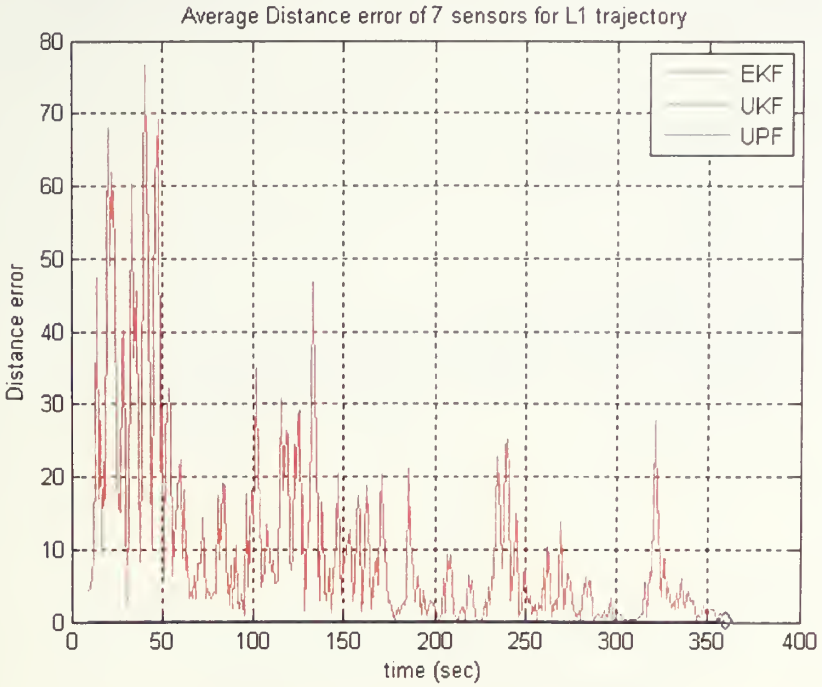

Figure 24. Average distance error of seven sensors for 1st lower trajectory

$$(\sigma_{range} = 10 meters, \sigma_{azimuth} = 1^0, \sigma_{elevation} = 1^0)$$

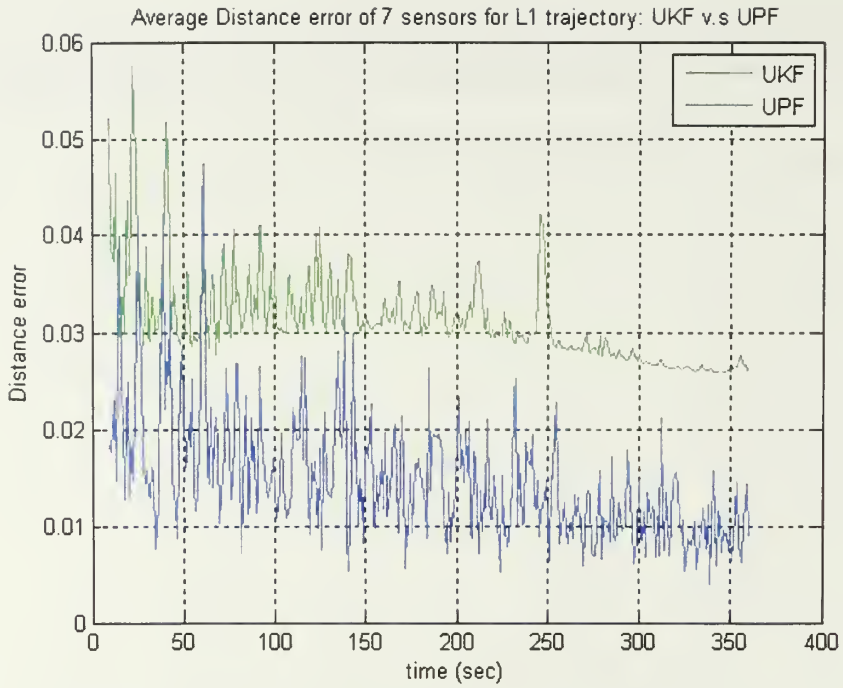

Figure 25. Distance error for the 1st lower stage trajectory (UKF vs. UPF)
$(\sigma_{range} = 10 meters, \sigma_{azimuth} = 1^0, \sigma_{elevation} = 1^0)$

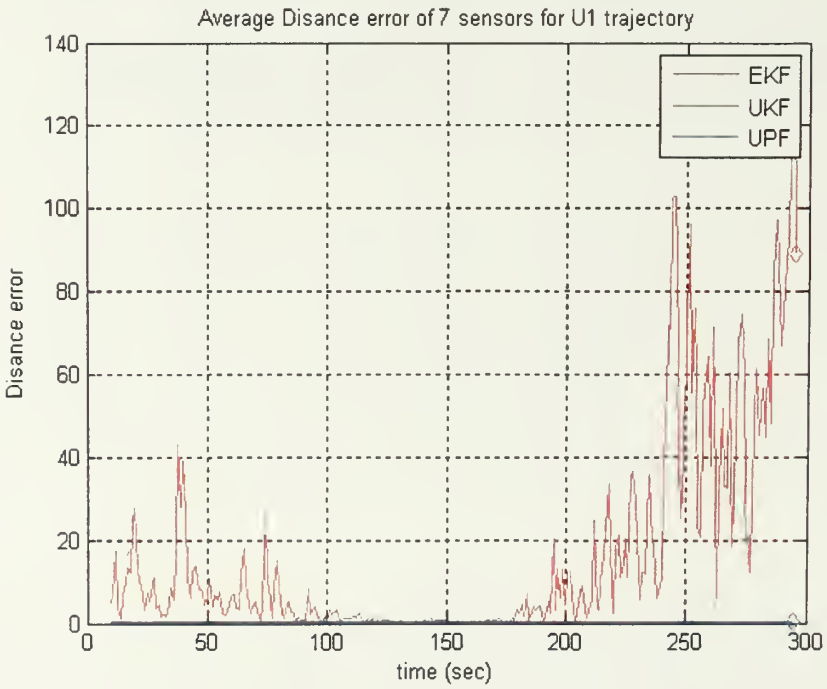

Figure 26. Average distance error of seven sensors for 1st upper trajectory
$(\sigma_{range} = 10 meters, \sigma_{azimuth} = 1^0, \sigma_{elevation} = 1^0)$

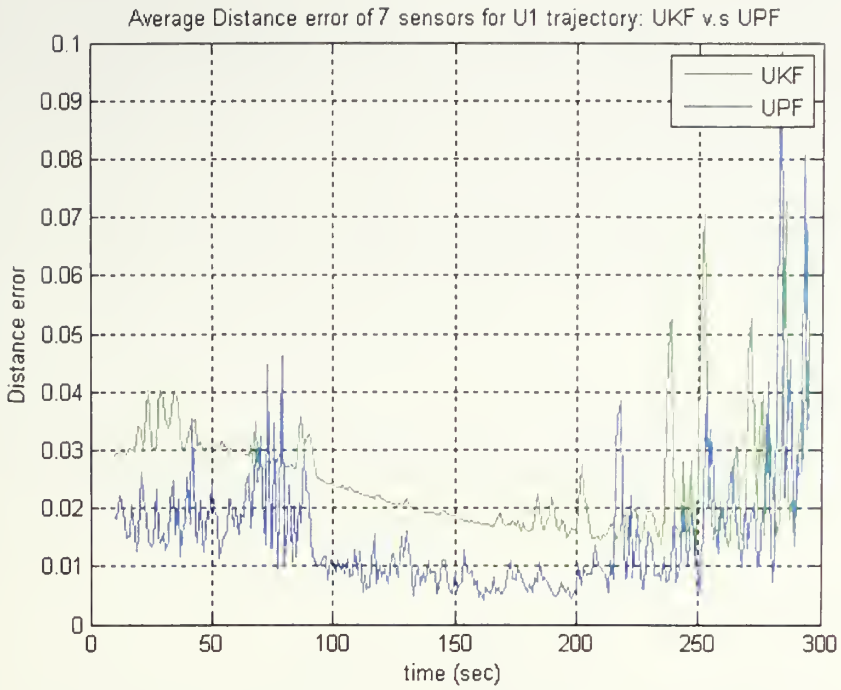

Figure 27. Distance error for the 1st upper stage trajectory (UKF vs. UPF)

$(\sigma_{range} = 10 meters, \sigma_{azimuth} = 1^0, \sigma_{elevation} = 1^0)$

Trajectory	EKF		UKF		UPF	
	Average	Last	Average	Last	Average	Last
U1	16.3177	89.0724	0.0244	0.0368	0.0149	0.0364
L1	10.2068	0.3315	0.0311	0.0259	0.0147	0.0089
U2	18.9756	160.6896	0.0281	0.0765	0.0182	0.0713
L2	25.2276	0.0115	0.1107	0.028	0.1007	0.026
U3	16.111	48.2313	0.0258	0.0301	0.0167	0.0447
L3	9.0777	3.0363	0.0324	0.0298	0.0168	0.0116
U4	28.1213	179.696	0.0339	0.1223	0.0197	0.0163

L4	2.6186	0.0115	0.0287	0.0214	0.0141	0.0055
U5	12.4995	36.0934	0.0302	0.0542	0.0206	0.0261
L5	15.1794	1.1139	0.0368	0.0332	0.0196	0.0251
U6	16.8421	38.389	0.0284	0.0353	0.0173	0.0141
L6	18.1476	7.1747	0.0385	0.0325	0.0206	0.0095

Table 6. Average distance error during the total time interval and average distance error at last time of three nonlinear filters for 12 trajectories
($\sigma_{range} = 10m, \sigma_{azimuth} = 1^{\circ}, \sigma_{elevation} = 1^{\circ}$)

C. MULTI-SENSOR FUSION

We compare the performance of the sensor fusion method using the EIF algorithm. We used the distance error, which is the square root of the sum for each coordinate distance error, as a measure to evaluate the performance. In this study, the measurement noise is chosen to have the following standard deviation:

- $\sigma_{range} = 10$ meters
- $\sigma_{azimuth} = 1^{\circ}$
- $\sigma_{elevation} = 1^{\circ}$

In order to evaluate the performance of the sensor fusion method, the Track Score Function (TSF) proposed in the previous master's theses is considered. The likelihood ratio for a combination of data, taking the priori probability data into account, is given as [19]:

$$\wedge(H_1, H_0) = \frac{P(D/H_1)Po(H_1)}{P(D/H_0)Po(H_0)} \square \frac{P_T}{P_F}$$

Where H_1 is the true target hypothesis, H_0 is the false target hypothesis, P_T is the probability of true target, P_F is probability of false target, $P(D \mid H_1)$ is the probability density function assuming that the H_1 is correct, $Po(H_1)$ is priori probability of H_1.

The performance of the tracking system should be independent of the behavior of the target. The system is evaluated for its ability to respond instantly to any change of the trajectory of the target either in velocity or direction.

Taking K scans of data, where the measurement error for one is not related to the error from the previous scan, the likelihood ratio ($\wedge$) can be given as the product of the likelihood ratio due to the kinematics ($\wedge_k$) and the likelihood ratio due to the signal ($\wedge_s$) is given:

$$\wedge(k) = \wedge_O \prod_{K=1}^{K} \wedge_k \wedge_s$$

Where

$$\wedge_O = \frac{P_o(H_1)}{P_o(H_0)}$$

In this experiment, we just considered a signal-related likelihood ratio. Based on likelihood ratio of the signal-related data, the track score is given by [18, Chapter 6]:

$$\wedge_s = \frac{P(Det / H_1)p(y_s / Det, H_1)}{P(Det / H_0)p(y_s / Det, H_0)} \qquad (1)$$

Taking both P_D and P_{FA} into account, (1) can be written as:

$$\wedge_s = \frac{P_D p(y_s / Det, H_1)}{P_{FA} p(y_s / Det, H_0)}$$

In the case of radar, when the SNR is available, the signal likelihood ratio is given by (1). The probability density function under H_1 us given by

$$p(y/H_1) = \frac{1+y/(1+2/\bar{\Theta})}{\left(1+\bar{\Theta}/2\right)^2}\exp\left[\frac{-y}{1+\bar{\Theta}/2}\right]$$

Where y is SNR data and $\bar{\Theta}$ is average of the SNR.

Probability density function under H_0 us given by

$$p(y/H_0) = \exp(-y)$$

Figures 25-47 show the distance error in the experiments that completed 30 Monte Carlo runs. The yellow line shows the distance error calculated from the sensor with the smallest error for three coordinates, where each sensor tracks the same target respectively. The blue line and green line are related to the TSF. The one results from a sensor with the biggest TSF at each time interval. The other one shows the results that fuse the top four sensors according to the TSF. Finally, the red line fuses all seven sensors. Figure 26 shows only the results of fusing all sensors and four sensors. Two figures of results are generated for each trajectory. The first figure shows the distance error graph of three fusion method and that of sensor with the smallest error at each time interval. The second one is to enlarge the result of only all sensor and four sensor fusion method to show the distance error. The diamond marker is designating the distance error at the final time.

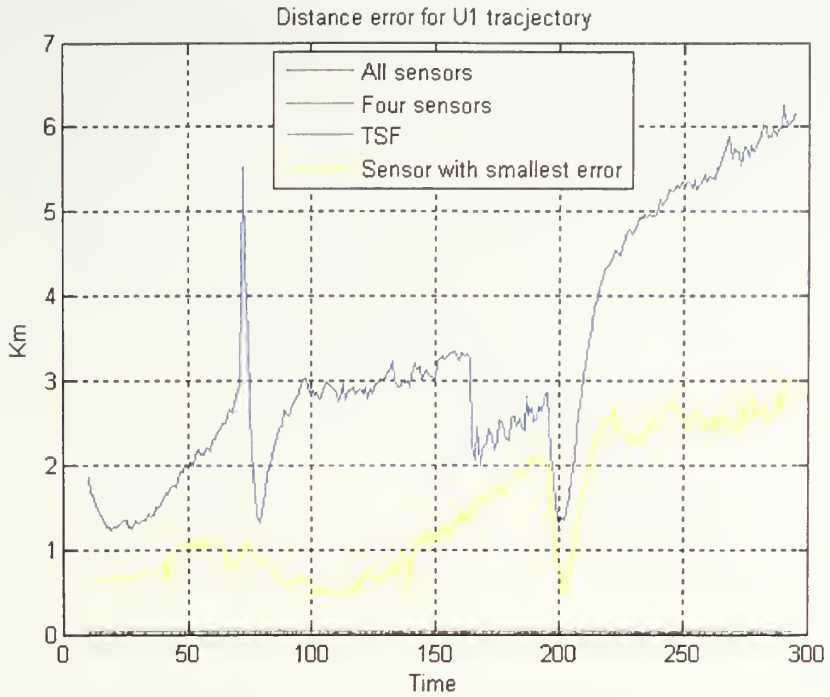

Figure 28. Distance error for the 1st upper stage trajectory (Three fusion methods and Sensor with smallest error)

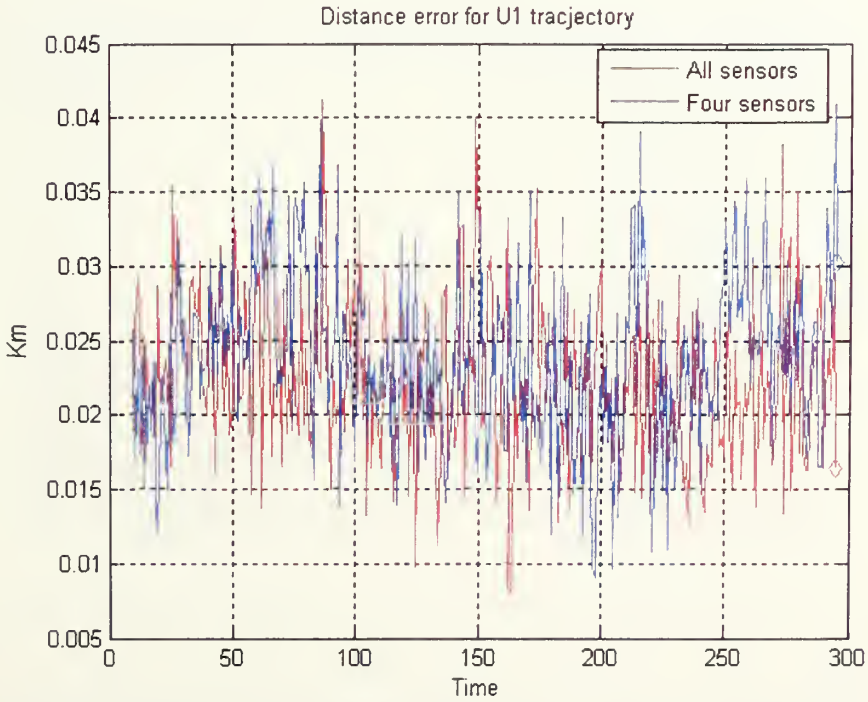

Figure 29. Distance error for the 1st upper stage trajectory (All sensors vs. Four sensors)

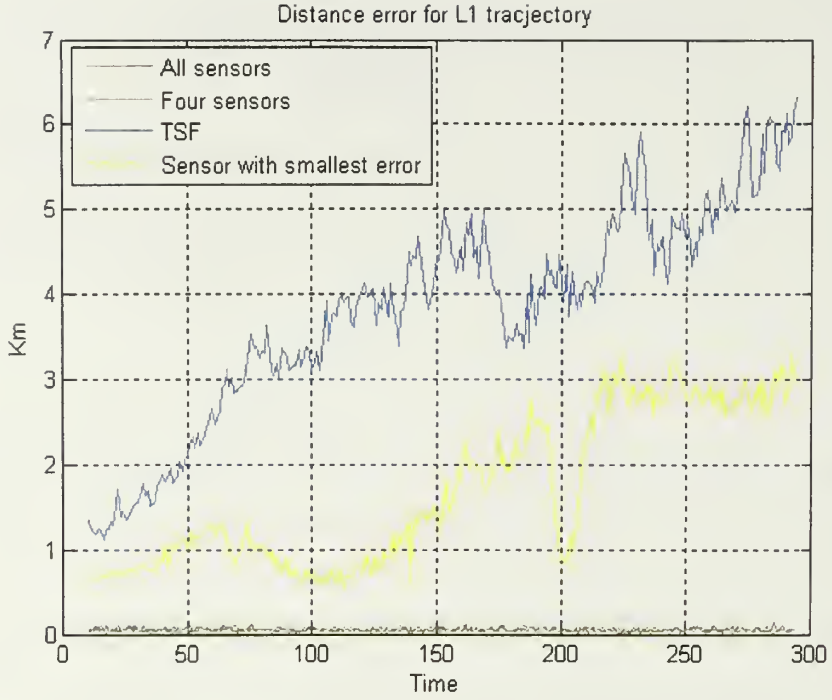

Figure 30. Distance error for the 1st lower stage trajectory (Three fusion methods and Sensor with smallest error)

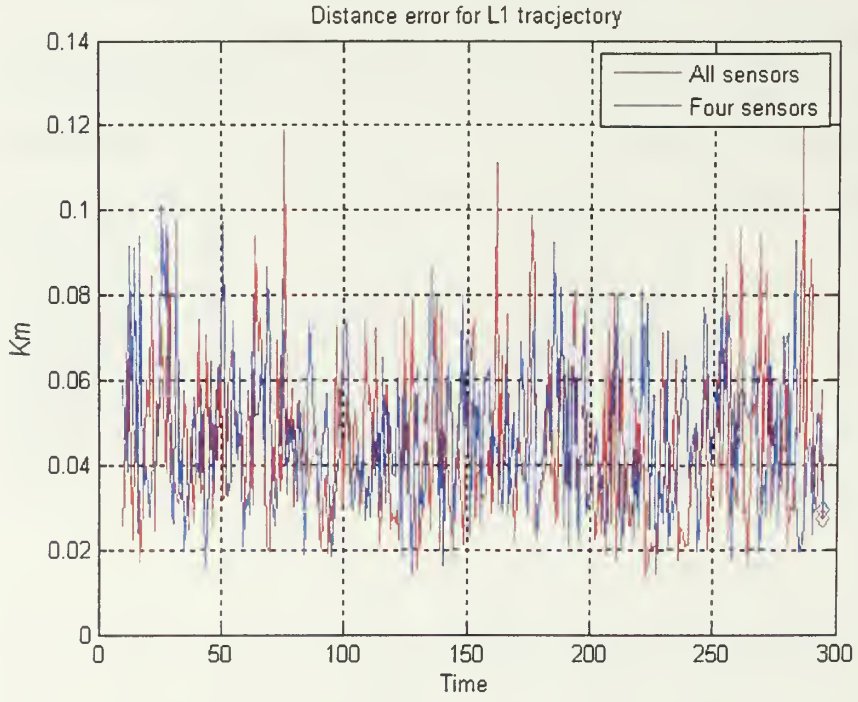

Figure 31. Distance error for the 1st lower stage trajectory (All sensors vs. Four sensors)

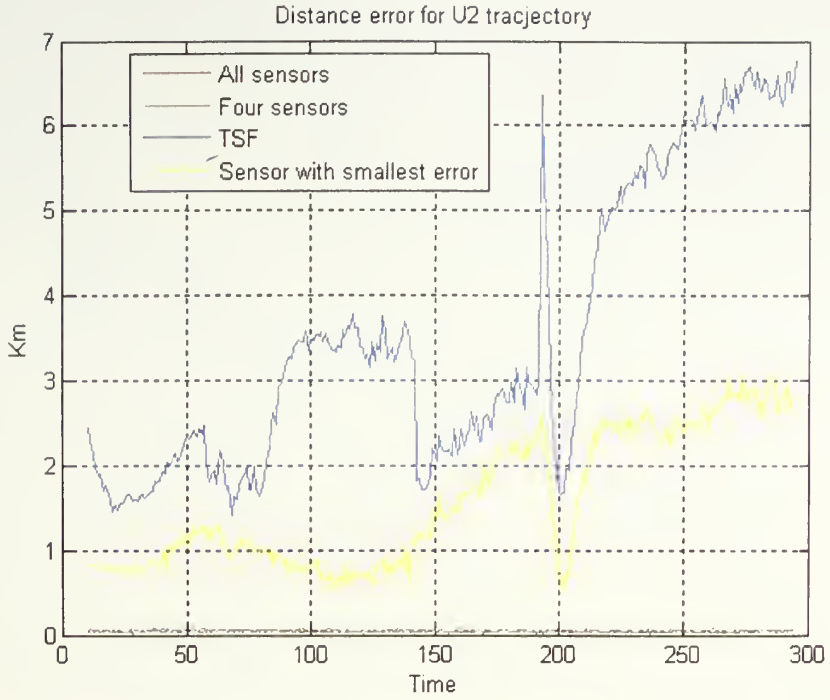

Figure 32. Distance error for the 2nd upper stage trajectory (Three fusion methods and Sensor with smallest error)

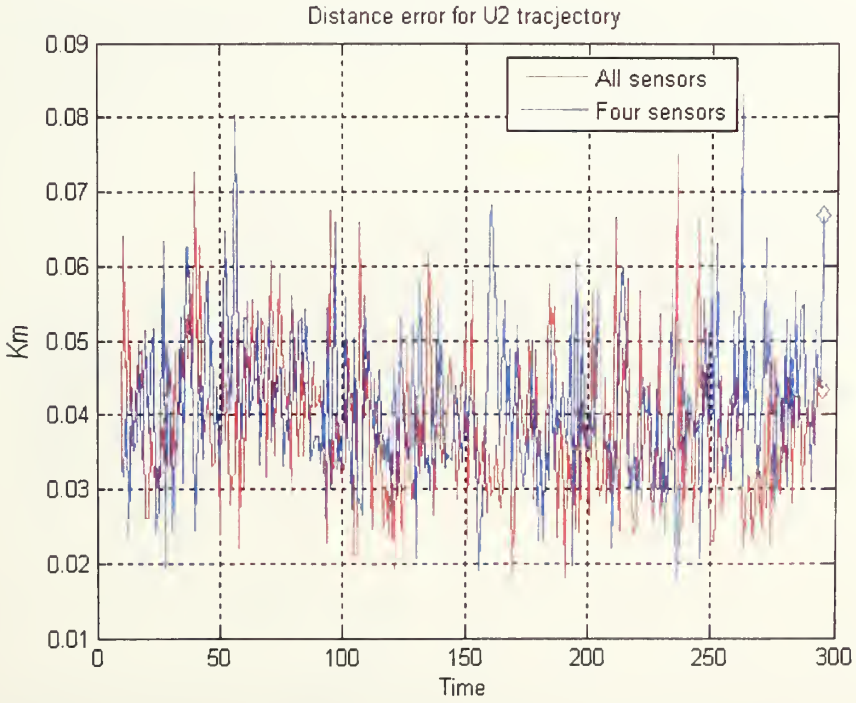

Figure 33. Distance error for the 2nd upper stage trajectory (All sensors vs. Four sensors)

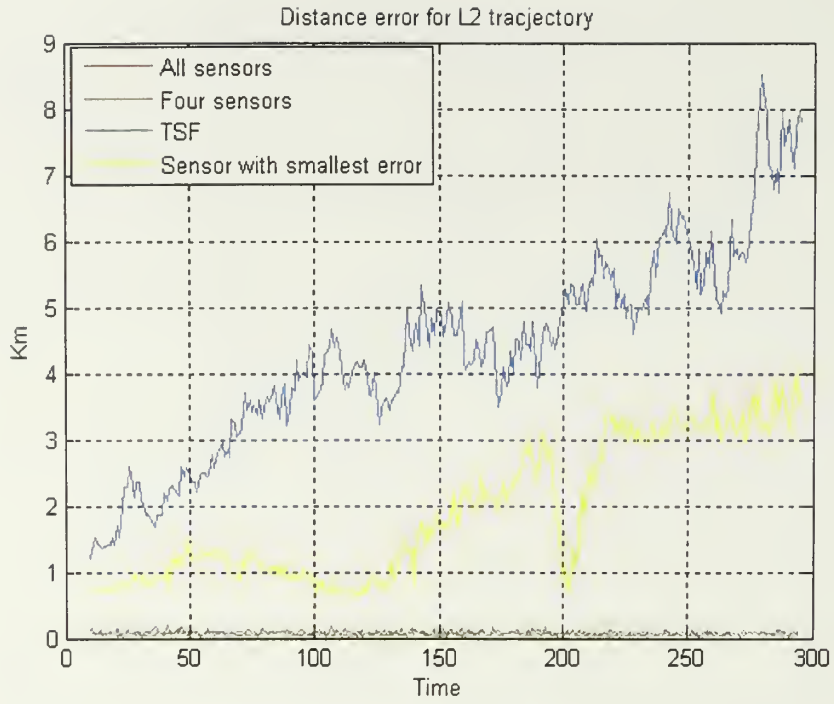

Figure 34. Distance error for the 2nd lower stage trajectory (Three fusion methods and Sensor with smallest error)

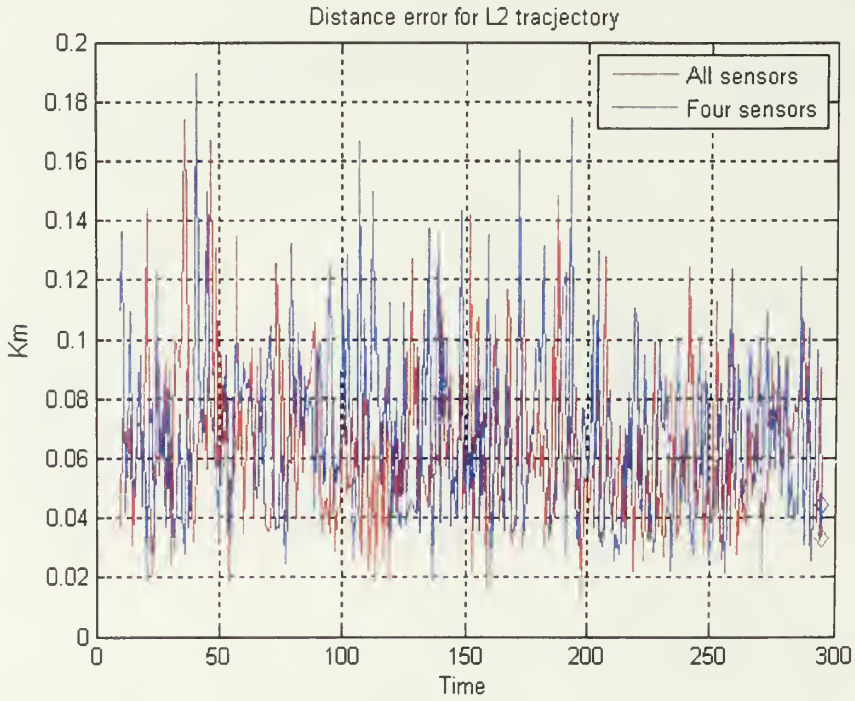

Figure 35. Distance error for the 2nd lower stage trajectory (All sensors vs. Four sensors)

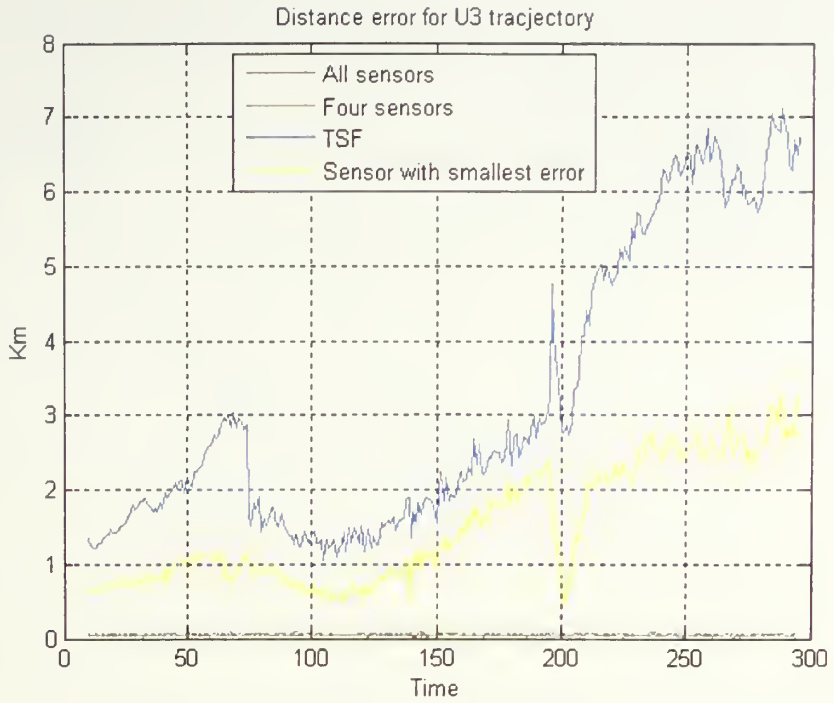

Figure 36. Distance error for the 3rd upper stage trajectory (Three fusion methods and Sensor with smallest error)

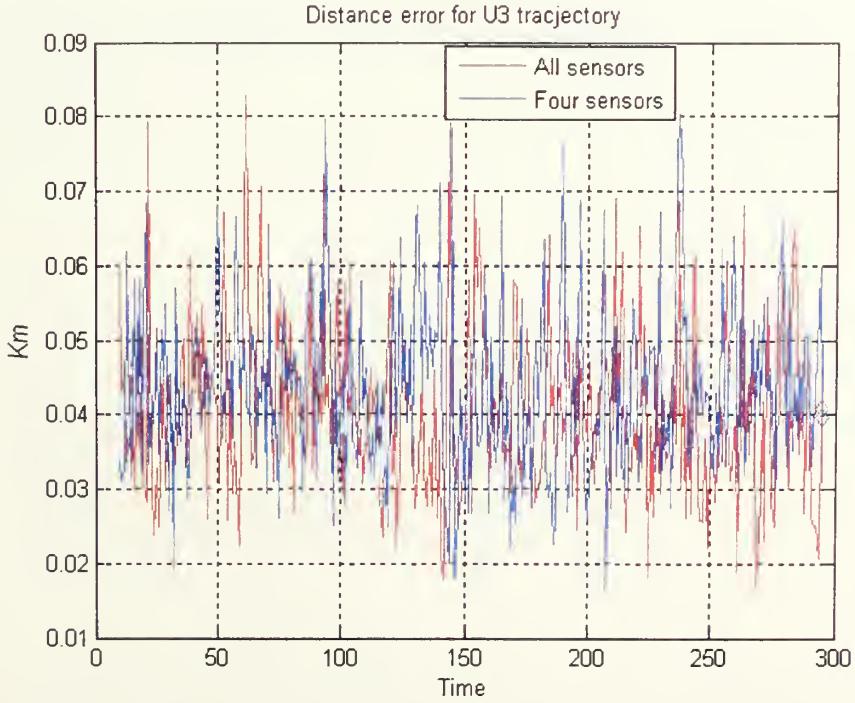

Figure 37. Distance error for the 3rd upper stage trajectory (All sensors vs. Four sensors)

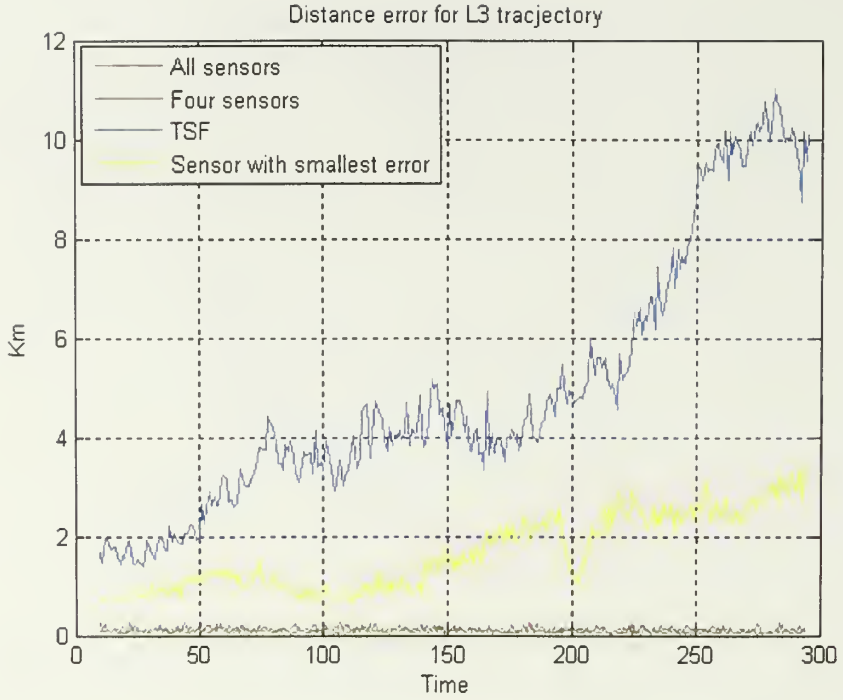

Figure 38. Distance error for the 3rd lower stage trajectory (Three fusion methods and Sensor with smallest error)

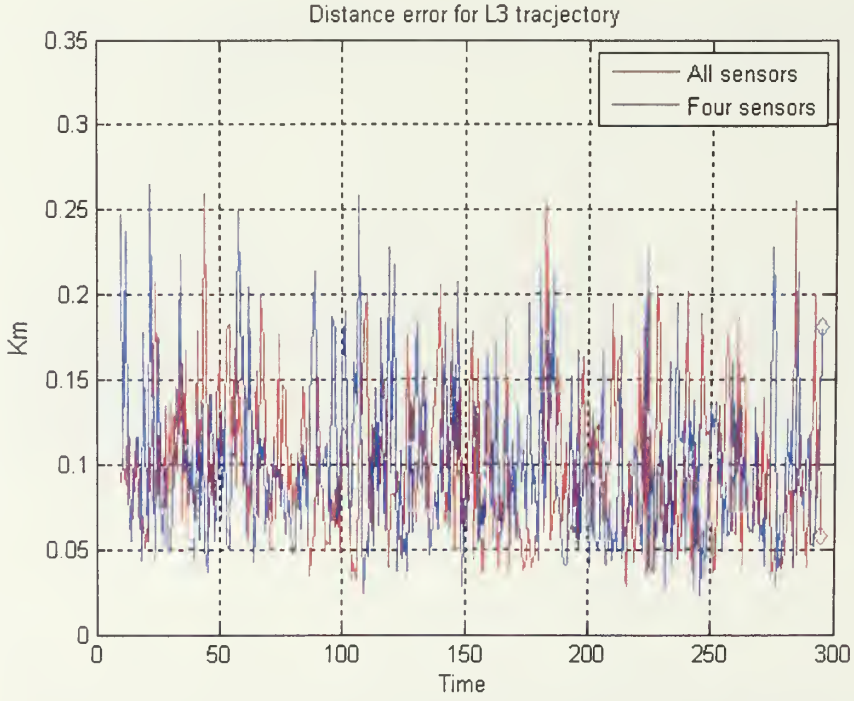

Figure 39. Distance error for the 3rd lower stage trajectory (All sensors vs. Four sensors)

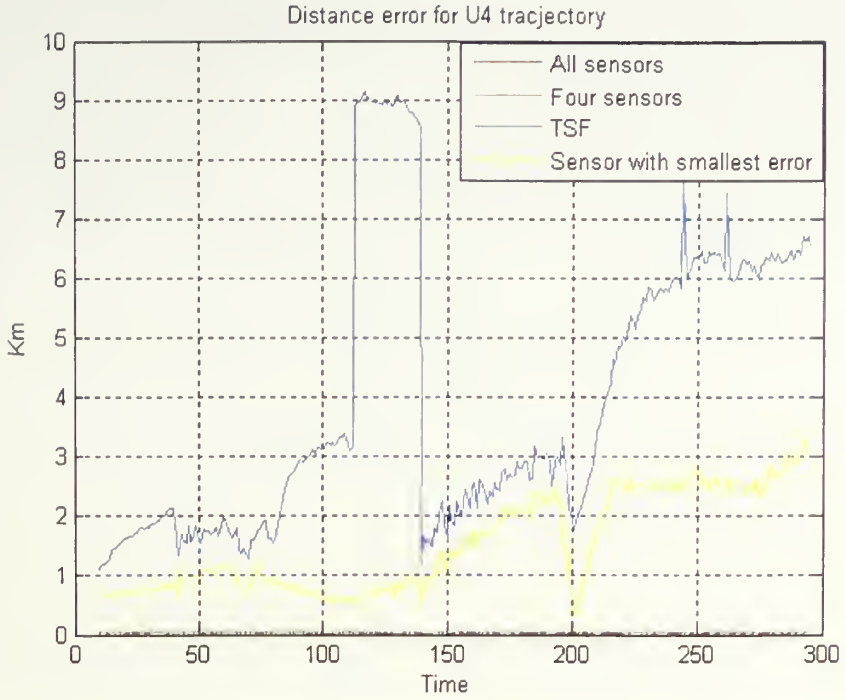

Figure 40. Distance error for the 4th upper stage trajectory (Three fusion methods and Sensor with smallest error)

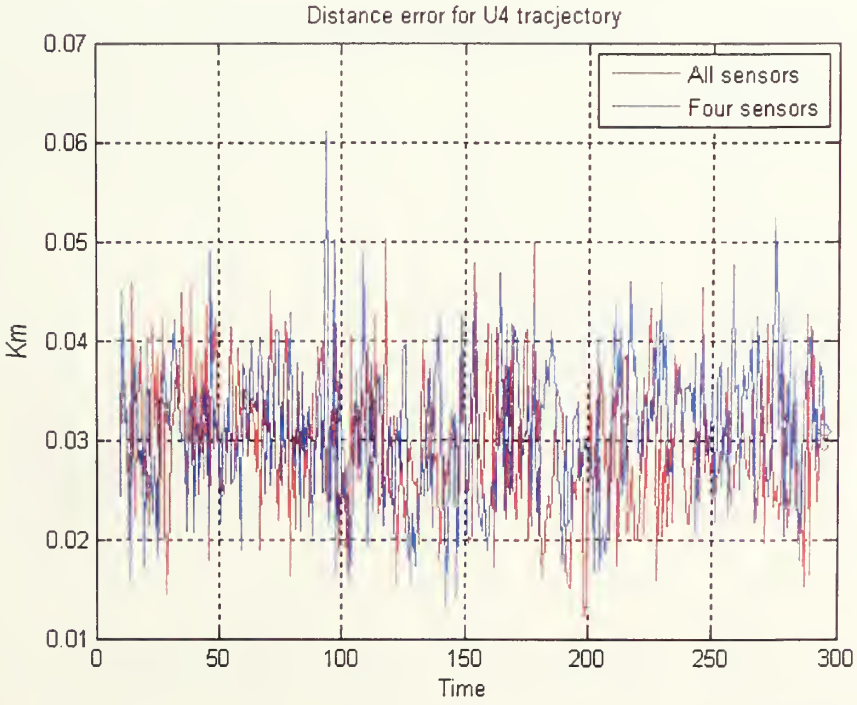

Figure 41. Distance error for the 4th upper stage trajectory (All sensors vs. Four sensors)

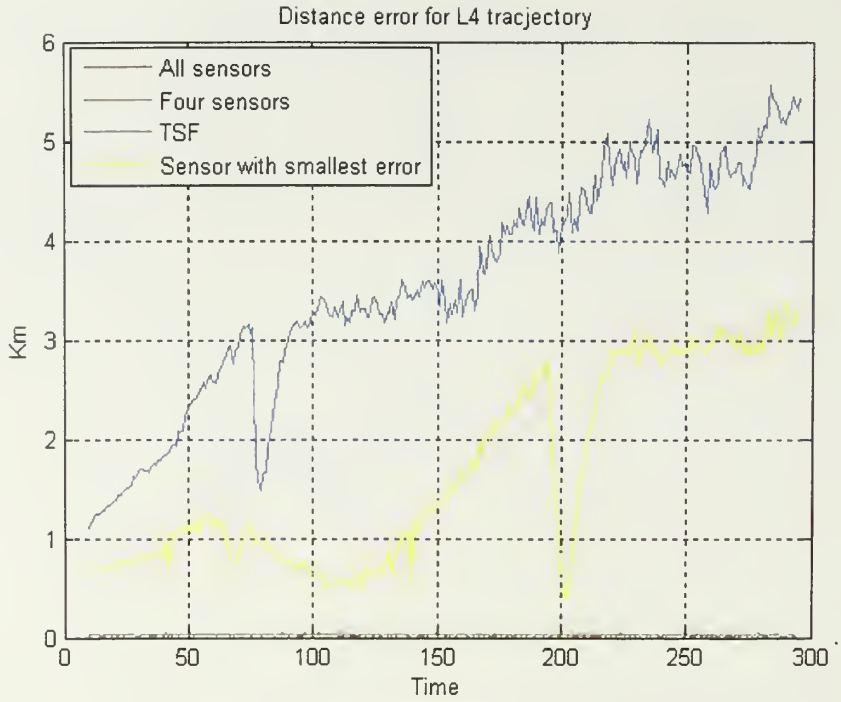

Figure 42. Distance error for the 4th lower stage trajectory (Three fusion methods and Sensor with smallest error)

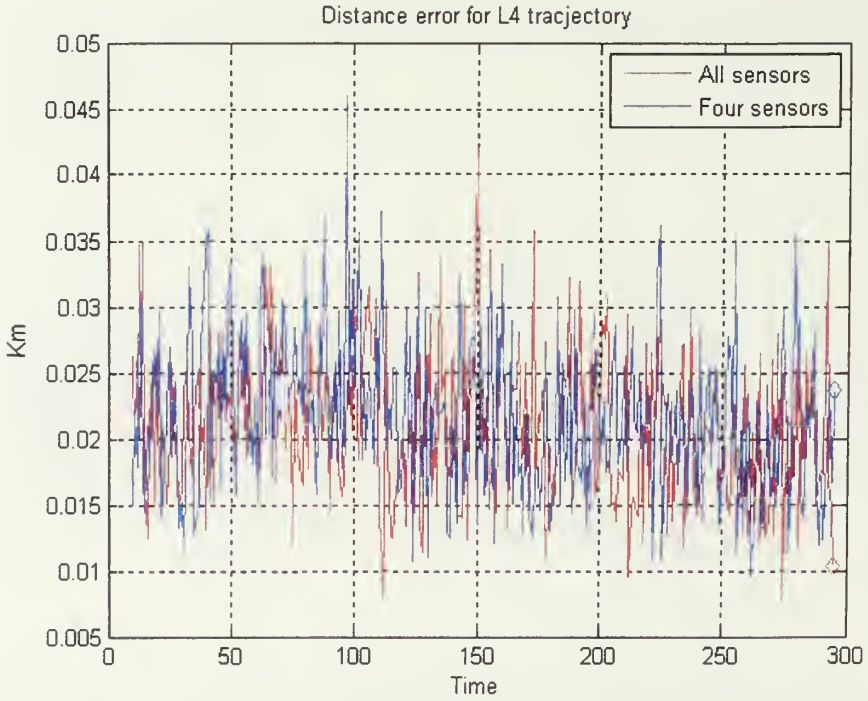

Figure 43. Distance error for the 4th lower stage trajectory (All sensors vs. Four sensors)

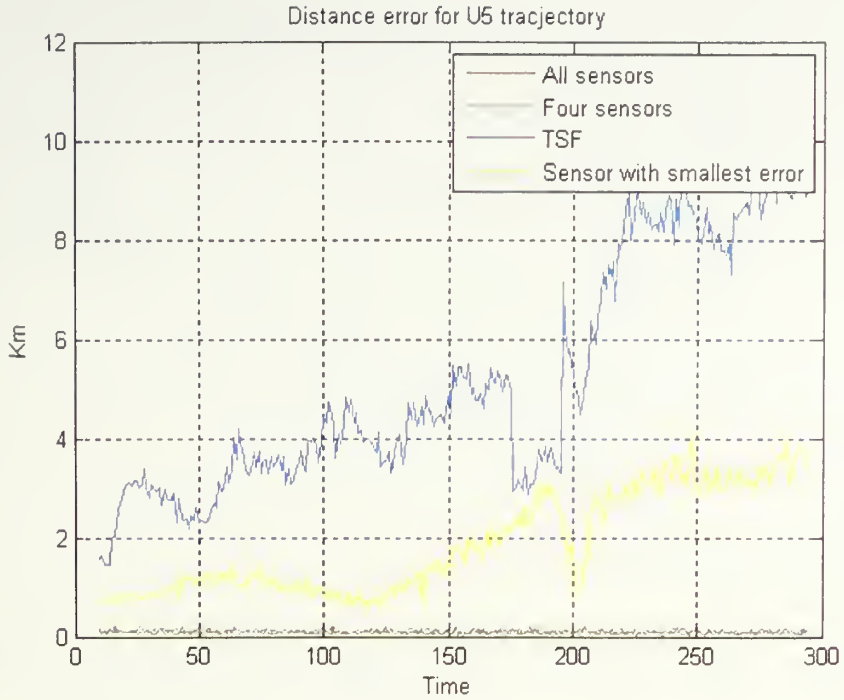

Figure 44. Distance error for the 5th upper stage trajectory (Three fusion methods and Sensor with smallest error)

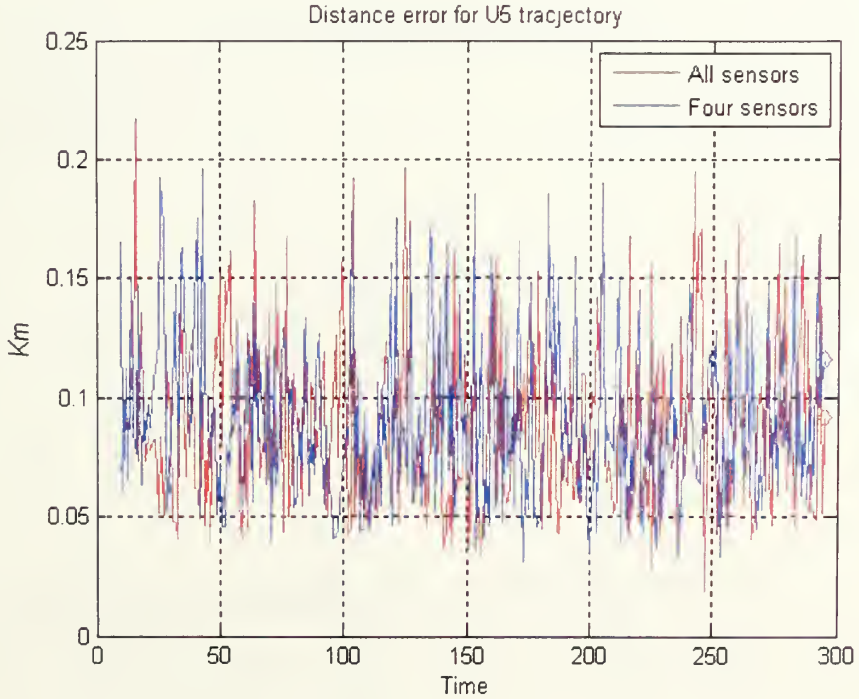

Figure 45. Distance error for the 5th upper stage trajectory (All sensors vs. Four sensors)

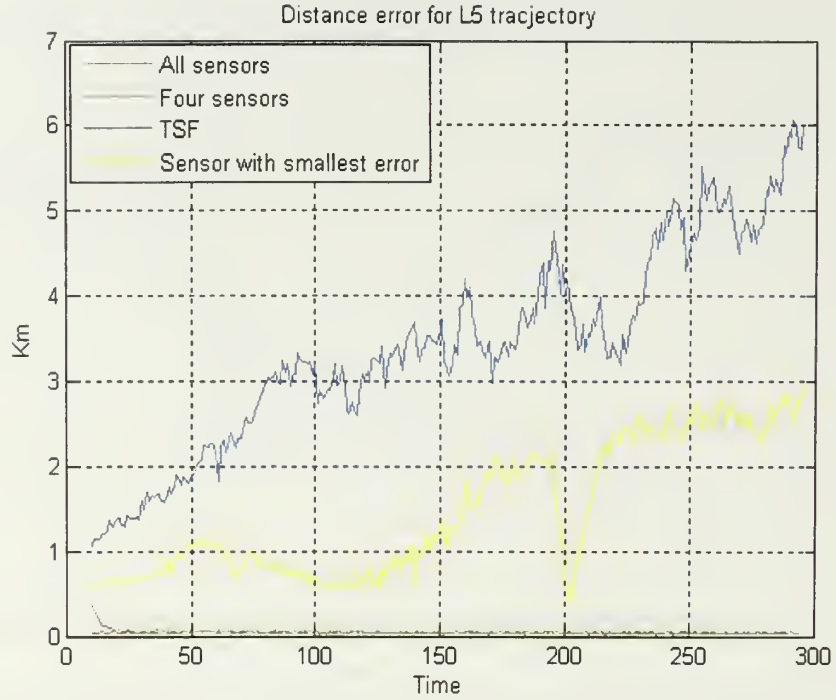

Figure 46. Distance error for the 5th lower stage trajectory (Three fusion methods and Sensor with smallest error)

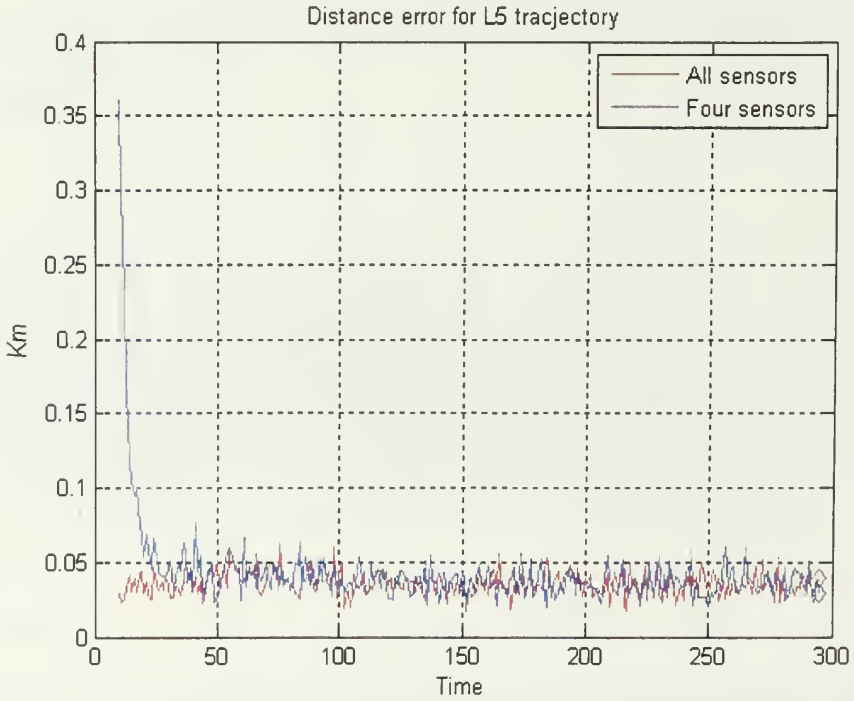

Figure 47. Distance error for the 5th lower stage trajectory (All sensors vs. Four sensors)

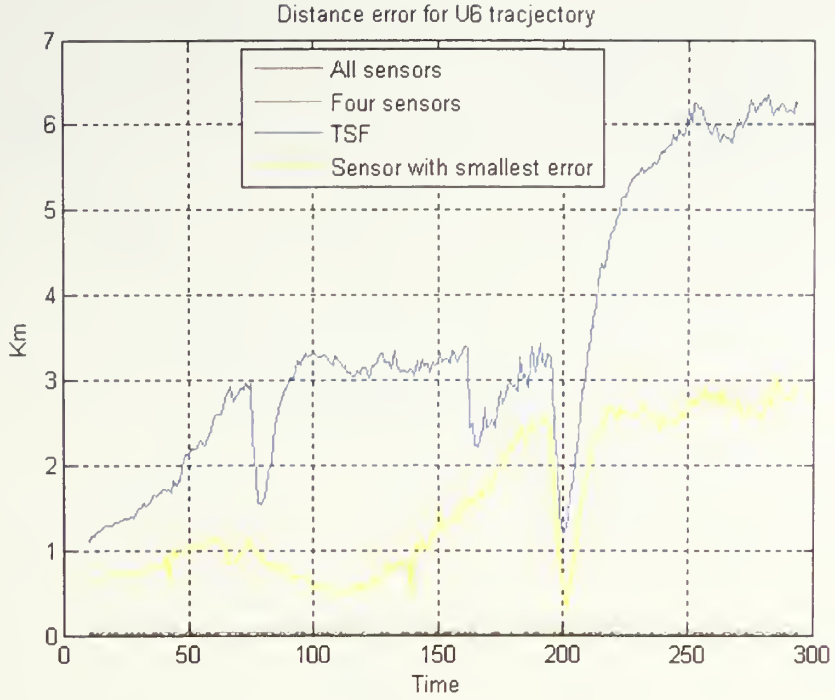

Figure 48. Distance error for the 6th upper stage trajectory (Three fusion methods and Sensor with smallest error)

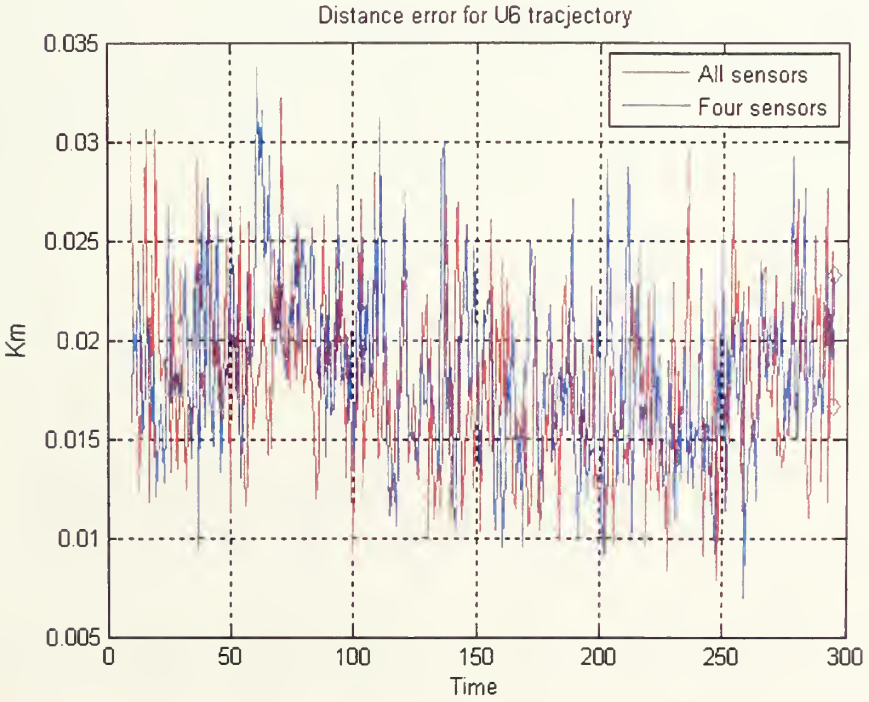

Figure 49. Distance error for the 6th upper stage trajectory (All sensors vs. Four sensors)

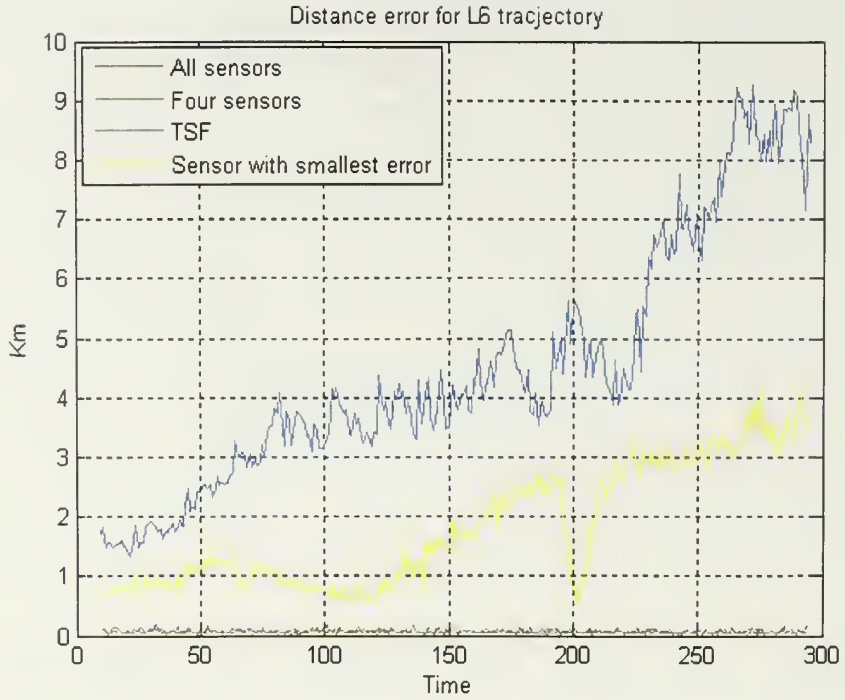

Figure 50. Distance error for the 6th lower stage trajectory (Three fusion methods and Sensor with smallest error)

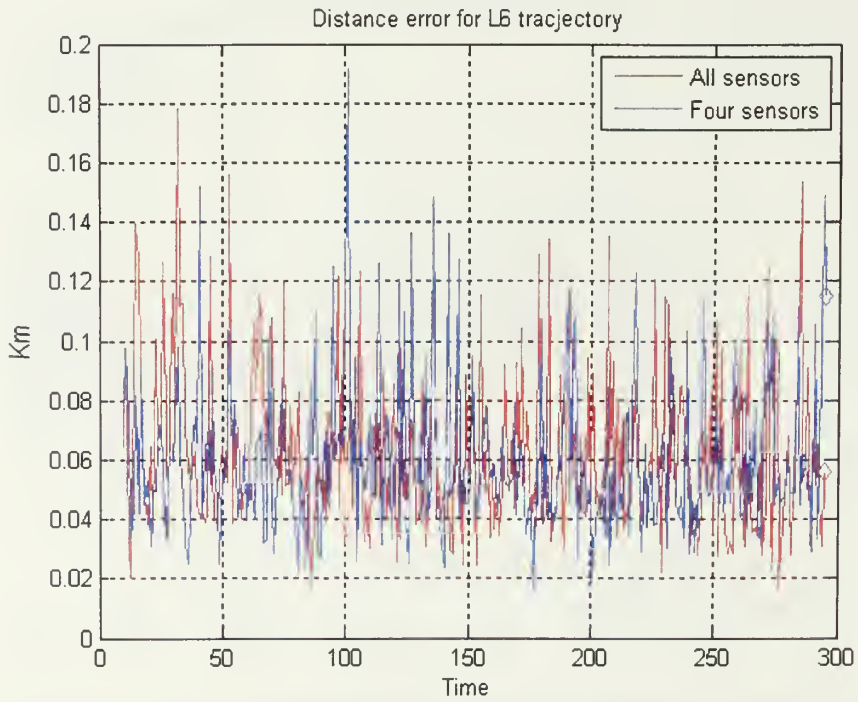

Figure 51. Distance error for the 6th lower stage trajectory (All sensors vs. Four sensors)

Tables 7 and 8 shows average distance error during total time period for three sensor fusion methods and sensor with the smallest error at each time interval. The result of fusion of all sensors or four sensors is better than the result of sensor with the best performance. Also, we can see that the performance of all sensor fusion is only slightly better than that of four sensor fusion. In conclusion, we can get good performance through sensor fusion based on the Extended Information Filter and can overcome the drawbacks of the Extended Kalman Filter in terms of performance and execution time.

Trajectory	All sensors	Four sensors	TSF (Track Score Function)	Sensor with the smallest error
U1	0.0225	0.0238	3.2469	1.4361
U2	0.0385	0.0412	3.5311	1.5719
U3	0.0406	0.0437	3.1981	1.5035
U4	0.0294	0.0311	3.9471	1.532
U5	0.0887	0.0913	5.1942	1.8709
U6	0.0177	0.0184	3.4899	1.5368
L1	0.0444	0.0468	3.8185	1.6734
L2	0.0644	0.0681	4.327	1.8802
L3	0.0986	0.0996	4.9319	1.6297
L4	0.0208	0.0218	3.5549	1.6639
L5	0.0359	0.0425	3.3999	1.4595
L6	0.0625	0.0598	4.4784	1.8106

Table 7. Average distance error of Monte-Carlo runs for 12 trajectories (Km)

Trajectory	All sensors	Four sensors	TSF (Track Score Function)	Sensor with the smallest error
U1	0.0165	0.0302	6.1695	3.0208
U2	0.0434	0.0667	6.7872	2.8399
U3	0.0407	0.0397	6.7577	3.3037
U4	0.0297	0.03087	6.5375	3.1721
U5	0.0915	0.1157	10.9499	3.5601
U6	0.0166	0.0232	6.2319	2.8892
L1	0.0273	0.0294	6.3306	2.8978
L2	0.033	0.0439	7.7776	3.4658
L3	0.0581	0.1807	10.1092	3.4979
L4	0.0104	0.0237	5.4433	3.3111
L5	0.0393	0.0281	5.9995	2.9402
L6	0.056	0.1149	8.249	3.4377

Table 8. Average distance error at the last time of Monte-Carlo runs for 12 trajectories (Km)

VI. CONCLUSION

This study has evaluated the performance of nonlinear filters, the Extended Kalman Filter, the Unscented Kalman Filter, the Particle Filter, and the Unscented Particle Filter and also developed a sensor fusion method to track multi-ballistic targets during boost phase using multi-sensors. This research makes use of the IMPULSE© simulation tool for generating ballistic missile flight profiles which serve as the test data for the tracking. We use the distance error as a performance measure. Our experimental results of comparing the performance of three nonlinear filters shows that the UPF has the best performance, but that the UKF is the best choice in terms of time, complexity, and performance, while the standard PF does not converge.

The fusion of data from several sensors provides significant advantages over single source data. In addition to the statistical advantage gained by combining same-source data, the use of multiple types of sensors may increase the accuracy with which a quantity can be observed and characterized. In this study, seven active ground-based radar sensors are used to track the ballistic missile during boost phase. We consider the Information Filter which is an efficient form to fuse the information from multiple sensors without information loss. To evaluate the performance of the Extended Information Filter, we compare with the track score function proposed in Patsikas's previous work in which the calculation of the track scoring function is to identify the sensor with the best track file. This study tells us that the result of multiple sensor fusion outperforms the performance of one sensor with the best performance and the information form of filter has an advantage in constructing a more efficient fusion architecture which is hierarchical and easily lends itself to decentralized processing.

VII. FUTURE WORK

In future work, the fusion architecture needs to be extended to a more efficient, fully decentralized form, so we may evaluate its performance and efficiency with those of the hierarchical fusion architecture. In addition, we should apply the Information Filter based on the Unscented Kalman Filter to track the Boost-phase ballistic missile, instead of the Extended Information Filter.

LIST OF REFERENCES

[1] A. Farina, B. Ristic, D. Benvenuti, "Tracking a ballistic Target: Comparison of Several Nonlinear Filters:, IEEE Transactions on AES, 38 (3), 854-867, July 2002.

[2] M . G. S. Bruno, A. Pavlov, "Improved Particle Filters for Ballistic Target tracking:, Optical Engineering, 43(6), 1-14, June 2004.

[3] B. G. Saulson, K. C. Chang, "Non-linear Estimation Comparison for Ballistic Target Tracking", ICASSO, II-705-708, 2004.

[4] G. M. Siouris, G. Chen, J. Wang, 'Tracking and Incoming Ballistic Missile Using an Extended Interval Kalman Filter", transactions on AES, 33(1), 232-239, Jan 1997.

[5] D. K. Barton et al., "Report of the American Physical Society Study Group on Boost-Phase Intercept Systems for National Missile Defense: Scientific and Technical Issues", Rev. Mod. Phys, 76, S1 2004.

[6] T. Lefebvre, H. Bruyninckx, J. De Schutter, "Kalman Filters for Nonlinear Systems: a Comparison of Performance", Internal report 01R033, Kaatholieke Universiteit Leuven, Oct. 2001.

[7] R. Van der Merwe, N. Feritas, A. Doucet, E. Wan, "The Unscented Particle Filter", Technical Report, CUED/F—INFENG/TR 380, Cambridge University Engineering Department, Aug. 2000. => unscented particle filteing

[8] S. J. Julier, and J. K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems, Proc. of AeroSense: The 11th international Symposium on Aerospace/Defence Sensing, Simulation and Controls, Orlando, Florida, Vol. Multi Sensor Fusion, Tracking and Resource Management II.

[9] S. J. Julier, and J. K. Uhlman, "Reduced sigma point filters for the propagation of means and covariance through nonlinear transformations", http://citeseer.nj.nec.com/julier98rediced.html.

[10] N.J Gordon, D.J. Salmond, and A.F.M. Smith, "Novel approach to Nonlinear/non-Gaussian Bayesian State Estimation", IEE Proceedings-F, 140(2), Apr. 1993.

[11] D. Crisan and A. Doucet, 'A Survey of Convergence Results on Particle Filtering Methods for Practitioners", IEEE Trans. On Signal Processing, 50(3), 736-746, Mar. 2002.

[12] A.G.O. Mutambara, Decentralized Estimation and Control for Multisensor systems. CRC Press, 1998.

[13] J. Manyika, S.Grime, and H. F. Durrant-Whyte, "A formally specified decentralized architecture for multi-sensor data fusion", In Transputing '91, pp.609-628, IOS Press, 1991.

[14] D.L. Hall, Mathematical Techniques in Multi-sensor Data Fusion, Artech House, London, 1992.

[15] S. Grime and Durrant-Whyte, "Data fusion in decentralized sensor networks", Control Eng. Practice, Vol.2, No.5, pp.849-863, 1994.

[16] B. Rakdham, "Efficient Multiple Hypothesis Track Processing of Boost-Phase Ballistic Missiles using IMPULSE-Generated Threat Models", MSEE Thesis, Naval Postgraduate School, CA, September 2006.

[17] E. Garrido, "Graphical user interface for a physical optics radar cross section prediction code", Master's Thesis, Naval Postgraduate School, Monterey, CA, 2000.

[18] S. Blackman and R. Popoli, Moderen Tracking System, Norward, NY: Artech House 1999 Publishers.

[19] P.F. Singer and D.M. Sasaki, "The Heavy tailed Distribution of a Common CFAR Detector", Signal and Data Processing of Small Targets, 1995, Proc. SPIE 2561, 1995, pp.124-140.

[20] D. Patsikas, "Track Score Processing of Multiple Dissimilar Sensors", MSEE Thesis, Naval Postgraduate School, CA, June 2007.

[21] D.B.Reid, "An algorithm for tracking multiple targets", IEEE Transaction on Automatic Control, Vol.AC-24, No.6, December 1979.

[22] V. Nagarajan, M.R.Chidambara, R.N.Sharma, "New Approach to Improved Detection and Tracking Performance in Track-While-Scan Radars", IEE Proceedings, Vol.134, Pt. F, No.1, February, 1987.

[23] I.J.Cox, S.L.Hingorani, "An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.18, No.2, February 1996.

[24] Y. Bar-Shalom and X. Li, Estimation and Tracking: Principles, Techniques, and Software, Artech House, Inc., Norwood, MA, 1993.

[25] L.D.Stone, C.A.Barlow, and T.L.Corwin, Bayesian Multiple Target Tracking, Artech House Inc. Maryland, 1999.

Initial Distribution List

1. Defense Technical Information Center — 2
8725 John J. Kingman Rd., STE 0944
Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library, Code 013 — 2
Naval Postgraduate School
Monterey, CA 93943-5100

3. Research Office, Code 09 — 1
Naval Postgraduate School
Monterey, CA 93943-5138

4. Phillip E. Pace, Professor
pepace@nps.edu
Department of Electrical and Computer Engineering — 1
Naval Postgraduate School
Monterey, CA 93943

5. James Bret Michael — 1
Bmichael@nps.edu
Department of Computer Science
Naval Postgraduate School
Monterey, CA 93943-5118

6. Dr. Carlo Kopp — 1
Carlo.kopp@iinet.net.au
Air Power Australia